ÓBUDAI EGYETEM
ÓBUDA UNIVERSITY

NEUMANN JÁNOS
INFORMATIKAI KAR

SZAKDOLGOZAT

OE-NIK	Hallgató neve:	**Tóth Áron**
2022	Hallgató törzskönyvi száma:	**T/008816/FI12904/N**

Óbudai Egyetem
Neumann János Informatikai Kar
Szoftvertervezés és -fejlesztés Intézet

SZAKDOLGOZAT FELADATLAP

Hallgató neve: **Tóth Áron**
Törzskönyvi száma: T/008816/FI12904/N
Neptun kódja: [redacted]

A dolgozat címe:

BartR, a közösségi cserekereskedelem-platform
BartR, the community barter trading platform

Intézményi konzulens: Kovács András, Molnár Attila
Külső konzulens:

Beadási határidő: 2022. május 15.

A záróvizsga tárgyai: Szoftvertechnológia és grafikus felhasználói interfész tervezése
Szoftvertervezés és -fejlesztés specializáció

A feladat

A feladat egy olyan közösségi cserekereskedelmi platform létrehozása, ahol az emberek nem csak eladhatják, de közvetlenül cserélhetik is megunt tárgyaikat, értékeiket. A dolgozat célja egyesíteni az eddigi online kereskedelmi oldalak erényeit és kiszűrni a felfedezett hiányosságokat. A hallgató hasonlítsa össze a hasonló témában írt rendszereket. A feladatban vizsgálja meg, hogy mitől lehet vonzó a célközönség számára egy ilyen webalkalmazás, mik azok a tényezők, amik motivációt jelenthetnek az oldal látogatására, a tárgyak cserére ajánlására. A rendszer biztosítson egy felhasználói felületet, ami a mai trendekhez hűen letisztult, könnyen navigálható, de megfelelő mennyiségű funkcionalitást magába tud sűríteni. A rendszer használjon modern adattároláshoz alkalmas adatbázist, amelyekben a különböző adatokat tárolja. Az üzleti logika megoldásához vizsgálja meg milyen technológiákkal lehetséges a megvalósítás. A rendszerhez készítsen okostelefonos alkalmazást is (Android vagy iOS platformra), illetve vizsgálja meg a platformfüggetlen lehetőségeket is. Vizsgálja meg annak a lehetőségét, hogy milyen felhőszolgáltatásban érdemes telepíteni. Végezzen továbbá elemzést a tesztelési lehetőségekről és készítse is el az egyes teszt típusokat és teszteseteket.

A dolgozatnak tartalmaznia kell:

- a szakirodalom részletes áttekintését és értékelését,
- a megoldandó feladat pontos leírását és részletes elemzését,
- a meglévő rendszerek elemzését,
- az alkalmazni kívánt technológiák elemzését, döntések indoklását,
- a rendszertervet,
- a megvalósítás pontos leírását,
- a tesztelési tervet, teszteredményeket, a fejlesztés során felmerült problémák elemzését,
- az elkészült rendszer felhasználási és továbbfejlesztési lehetőségeit.

ÓBUDAI EGYETEM
Neumann János Informatikai Kar
1.
Budapest

.....
Dr. Vámossy Zoltán
intézetigazgató

A szakdolgozat elévülésének határideje: **2024. május 15.**
(OE TVSz 55.§ szerint)

A dolgozatot beadásra alkalmasnak tartom:

.....
külső konzulens

.....
intézményi konzulens

ÓBUDAI EGYETEM
ÓBUDA UNIVERSITY

Neumann János Informatikai Kar

HALLGATÓI NYILATKOZAT

Alulírott hallgató kijelentem, hogy a szakdolgozat / diplomamunka saját munkám eredménye, a felhasznált szakirodalmat és eszközöket azonosíthatóan közöltem. Az elkészült szakdolgozatomban / diplomamunkámban található eredményeket az egyetem és a feladatot kiíró intézmény saját céljára térítés nélkül felhasználhatja.

Budapest, 2022. 12. 07.

.....

hallgató aláírása

9. melléklet

ÓBUDAI EGYETEM
ÓBUDA UNIVERSITY

Neumann János Informatikai Kar

KONZULTÁCIÓS NAPLÓ

Hallgató neve: Tóth Áron

Neptun Kód: [redacted]

Tagozat: nappali

Telefon: [redacted]

Levelezési cím (pl.: lakcím): [redacted]

Szakdolgozat címe magyarul:

BartR, a közösségi cserekereskedelem-platform

Szakdolgozat címe angolul:

BartR, the community barter trading platform

Intézményi konzulens: Kovács András, Molnár Attila

Külső konzulens:

Kérjük, hogy az adatokat nyomtatott nagy betűkkel írja!

Alk.	Dátum	Tartalom	Aláírás
1.	2022. 02. 20	Szakdolgozat téma tervezése	
2.	2022. 03. 18	Irodalomjegyzék áttekintése	
3.	2022. 04. 15	Tervezési fázis kiértékelése	
4.	2022. 05. 10	Prezentációs próba	

A Konzultációs naplót összesen 4 alkalommal, az egyes konzultációk alkalmával kell láttamoztatni bármelyik konzulenssel.

A hallgató a Szakdolgozat I. (BSc) tantárgy követelményét teljesítette, beszámolóra bocsátható.

.....

Intézményi konzulens

Budapest, 2022. 05. 20

9. melléklet

ÓBUDAI EGYETEM
ÓBUDA UNIVERSITY

Neumann János Informatikai Kar

KONZULTÁCIÓS NAPLÓ

Hallgató neve: Tóth Áron

Neptun Kód: [redacted]

Tagozat: nappali

Telefon: [redacted]

Levelezési cím (pl.: lakcím): [redacted]

Szakdolgozat címe magyarul:

BartR, a közösségi cserekereskedelem-platform

Szakdolgozat címe angolul:

BartR, the community barter trading platform

Intézményi konzulens: Kovács András, Molnár Attila

Külső konzulens:

Kérjük, hogy az adatokat nyomtatott nagy betűkkel írja!

Alk.	Dátum	Tartalom	Aláírás
1.	2022. 09. 20	Kutatási fázis áttekintése	
2.	2022. 10. 18	Implementáció tervezése	
3.	2022. 11. 15	Tesztelés, eredmények áttekintése	
4.	2022. 11. 28	Prezentációs próba	

A Konzultációs naplót összesen 4 alkalommal, az egyes konzultációk alkalmával kell láttamoztatni bármelyik konzulenssel.

A hallgató a Szakdolgozat II. (BSc) tantárgy követelményét teljesítette, védésre bocsátható.

.....

Intézményi konzulens

Budapest, 2022. 11. 28

TARTALOMJEGYZÉK

1 ABSZTRAKT

A feladat egy olyan közösségi cserekereskedelmi platform létrehozása, ahol az emberek nem csak eladhatják, de közvetlenül cserélhetik is megunt tárgyaikat, értékeiket. A dolgozat célja egyesíteni az eddigi online kereskedelmi oldalak erényeit és kiszűrni a felfedezett hiányosságokat.

A hallgató hasonlítsa össze a hasonló témában írt rendszereket. A feladatban vizsgálja meg, hogy mitől lehet vonzó a célközönség számára egy ilyen webalkalmazás, mik azok a tényezők, amik motivációt jelenthetnek az oldal látogatására, a tárgyak cserére ajánlására. A rendszer biztosítson egy felhasználói felületet, ami a mai trendekhez hűen letisztult, könnyen navigálható, de megfelelő mennyiségű funkcionalitást magába tud sűríteni. A rendszer használjon modern adattároláshoz alkalmas adatbázis, amelyekben a különböző adatokat tárolja. Az üzleti logika megoldásához vizsgálja meg milyen technológiákkal lehetséges a megvalósítás. A rendszerhez készítsen okostelefonos alkalmazást is vagy Android vagy iOS platformra, illetve vizsgálja meg a cross-platform[1] lehetőségeket is. Vizsgálja meg annak a lehetőségét, hogy milyen felhőszolgáltatásban érdemes telepíteni.

Végezzen továbbá elemzést a tesztelési lehetőségekről és készítse is el az egyes teszt típusokat és teszteseteket.

[1] Több platformra való megvalósítás

2 BEVEZETÉS

A gazdaság mindig is meghatározó szerepet töltött be mindennapjainkban. A kezdetben csak cserekereskedelem volt: köveket cseréltünk kövekre, szerszámokat szőrmékre. Ez viszont ekkor még nagyon körülményes volt: Sokszor előfordult, hogy akár mi, akár a másik fél nem akart cserélni az adott értéktárgyra, mert egyszerűen nem tetszett neki vagy nem arra volt szüksége. Nehéz volt azt is meghatározni, hogy mennyire képviselnek egyenlő értéket a cserére szánt értékek. Ezen folyamat persze hamar felfejlődött és átalakult: létrejöttek az általános csereeszközök és a pénz. Így lehetőség nyílt arra, hogy ne kelljen azzal foglalkozni, milyen tárgyra lenne szüksége a másik személynek vagy hogy ugyanannyit érnek-e a cserére ajánlott tárgyak, egyszerűen ki lehetett őket fizetni.[1]

Manapság talán ki lehet mondani, hogy ennek az evolúciónak köszönhetően kihaló félben van a cserekereskedelem, inkább az elmaradt régiókban van még jelen. El kell ismerni, hogy a kereskedelem mostani formája kényelmesebb és modernebb, de megvannak a hátulütői. A jelen fogyasztói társadalom velejárója, hogy rengeteg dolgunk van, amikre egyáltalán nincs szükségünk, nem használjuk és csak porosodnak. Ahhoz, hogy ezektől megváljunk, a leggyakoribb módszer az eladás: választunk egy online apróhirdetési oldalt, a tárgy feltöltésre kerül és több-kevesebb eséllyel meg is veszik.

Határon belül és kívül három, piacot uraló apróhirdetési oldal létezik. Országunkban a Jófogás, a Facebook Marketplace és a Vatera uralkodik, amíg nemzetközi szinten a Marketplace mellett a Craigslist és az Ebay. Mivel minden nap használom őket, lehetőségem adódott arra, hogy felfedezzem a fenti platformok erényeit és hiányosságait, illetve hibáit. Szakdolgozatomban megvizsgálom őket, összehasonlítom és a kinyert tanulságokból megtervezek majd létrehozok egy új platformot, ami nem csak kiküszöböli hibáikat, hanem tovább is gondolja az online apróhirdetések rendszerét és új funkciókkal bővíti azt.

Bár a technológia előrehaladtával háttérbe szorult a cserekereskedelem, azért ma is megvan a helye: Erre épül az alapötletem. R engeteg olyan szituáció van, ahol sokkal kényelmesebb lenne akár egy-az-egyben, akár értékegyeztetéssel cserélni. Mivel erre a jelenlegi kereskedelmi platformok nem adnak közvetlen lehetőséget, ez lesz az általam elkészítendő oldal mozgatórugója; emellé olyan funkciókat fogok beágyazni, amik megkönnyítik a cserék teljes folyamatát és motivációt adnak további üzletkötésekre.

3 IRODALOMKUTATÁS

Az irodalomkutatásom során a legnagyobb hangsúlyt a meglévő és piacvezető apróhirdetési oldalak vizsgálata fogja kapni: Ki kell elemezni, mi teszi őket sikeressé, mik azok a szolgáltatások, amik a felhasználók szempontjából kiemeli őket a többi platform közül és fenntartja az érdeklődést. Össze kell gyűjteni és rangsorolni a legfontosabb funkciókat és úgy implementálni őket, hogy az új platform többet nyújthasson, vonzóbb legyen, a megtalált hibák javításra kerüljenek és emellett felhasználóbarát maradjon. A kutatás során kellő figyelmet fordítok a bel- és külföldi társadalmi és technológiai eltérésekre, ugyanis nagyban befolyásolják a felhasználók igényeit.

3.1 HASONLÓ RENDSZEREK

Először a jelenleg is működő, piacot uraló platformokat vizsgálom meg. Az elemzés során először az adott rendszerek rövid történeteit, majd felhasználói felületét és különlegességeit vetem górcső alá.

3.1.1 JÓFOGÁS

Hazánk második legnépszerűbb apróhirdetési oldala. Felépítése egyszerű, sallangoktól mentes.

1. ábra – Jófogás

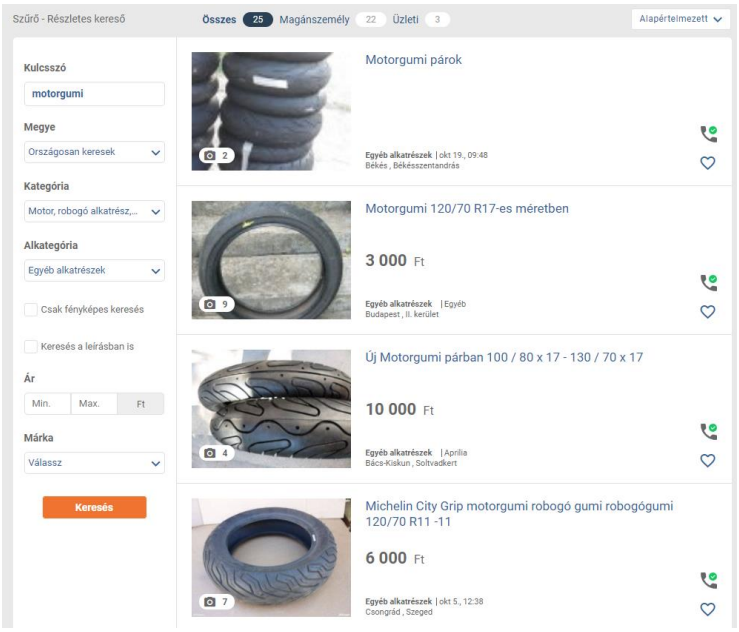
A webalkalmazásoknál az egyik legfontosabb alapkövetelmény, hogy az oldalt használók eltárolhassák adataikat, rögzíthessék az azokban végbement változásokat. A felhasználói fiók létrehozásának folyamatának minél rövidebbnek és egyszerűbbnek kell lennie, ugyanis a legtöbb látogató itt fordul vissza, ha azt látja, hogy túl sok időt vesz igénybe. Erre jó módszer, ha egy már meglévő Google vagy Facebook fiók csatolásával elkészül a fiók, de sokszor az is elég, ha kellően egyszerű a fiókkészítés és bejelentkezés.

Új felhasználó regisztrálásának szempontjából a Jófogás viszi a prímet, itt egy felhasználónév, e-mail cím és jelszó megadásával - majd annak megerősítésével - már indulhat is a hirdetésfeltöltés. Ez adja a legszabadabb megoldást, de ez a legkevésbé hatékony a csalók kiszűrésében is.

A felhasználói felület bár intuitív és letisztult, mégis lassan már elavultnak mondható. Az elemek statikusak, nincsenek felugró gombok vagy ablakok, amik megkönnyíthetik az oldal kezelését.

Itt a legnagyobb - mind a négy platformon előforduló, de itt a leginkább - problémát a termékek kategorizálásában véltem felfedezni. Számos termék van, amihez ha pontos megnevezés vagy hívószó nem ismert, nagyon nehéz rátalálni. Például, ha motorkerékpár-gumit keresnénk, két lehetőség van: az „autógumi” vagy az „egyéb motorkerékpár-alkatrészek” kategória. Mivel egyikbe sem illik bele, gyakran egy

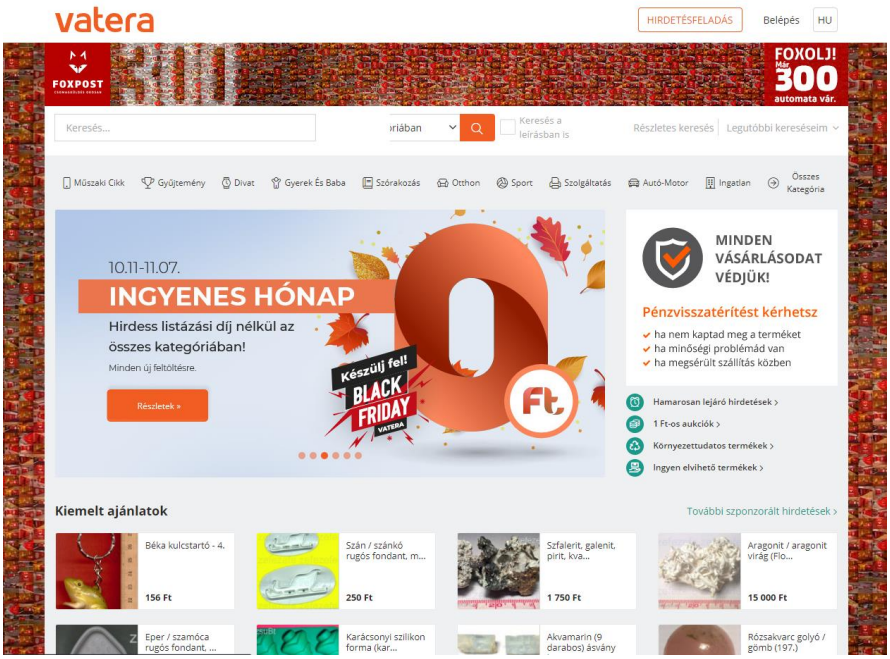
harmadik helyre töltik fel a felhasználók a hirdetést, aminek az az eredménye, hogy a termék gyakorlatilag eladhatatlan lesz.



2. ábra – Találatok listázása a Jófogáson

3.1.2 VATERA

Országunk legnépszerűbb online piactere. 2000 óta működik. Kifejezett előnye a jófogással szemben az, hogy lehetőséget kínál aukciók létrehozására, ahol nem csak fix áron, hanem licitálással is megvásárolhatók a tárgyak.



3. ábra - Vatera

Az oldalon lehetőség van Facebookos regisztrációra, ami nem csak gyors, hanem biztonságos is.

A Jófogással szemben itt nem csak a vevők az eladókat, hanem az eladók is értékelhetik a vevőket. Ez az alapvető közösségi funkción kívül segít a csalók kiszűrésében is: a megbízhatatlan vevők rossz értékelése előre jelezheti, hogy fokozott óvatossággal kell velük üzletet kötni.

Egy webalkalmazás viszont lehet bármilyen hatékony vagy funkciókkal dúsított, megfelelő felhasználói felület (továbbiakban: front-end) nélkül a konkurencia majdnem mindig vonzóbb lesz. Ilyen szempontból hazánkban a Vatera vezet; nemrég átalakították a felületüket, ami így már illeszkedik a 2021-es elvárásokhoz: reszponzív, gyors és letisztult.

A megfelelő marketing hiányán kívül a legnagyobb hiányosságot abban látom, hogy nincsenek megfelelő szűrési feltételek. Számos termékre konkrét megnevezés nélkül nem tudunk rátalálni, mert nincs lehetőség a találatok paraméter alapú szűrésére.

3.1.3 CRAIGLIST

Biztonsággal kijelenthető, hogy a legrégebbi, az évek alatt a legkevesebb változást megélt és ezzel együtt Amerikában a legnépszerűbb apróhidetési platform. Az idők folyamán monolitikussá nőtt, rengeteg különböző dolog megtalálható lett rajta állásajánlatoktól, eladó házakon és koncerteken át mindenféle szolgáltatásig. Mintapéldája annak, hogy

mennyire fontos az értékesítésre szánt tárgyak megfelelő kategorizálása, hiszen sikerét ennek köszönheti. Néhány esetben

4. ábra - Craigslist

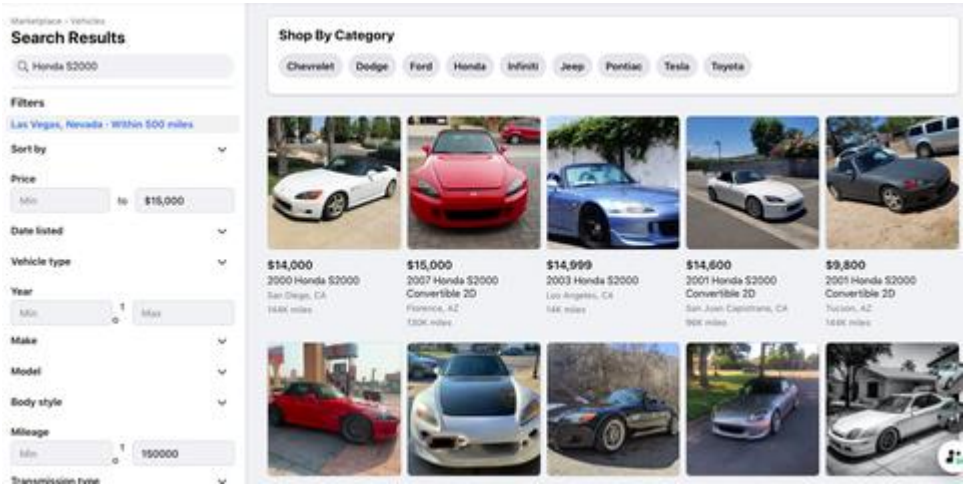
Felhasználói felülete kifejezetten elavult, mára már szinte megmosolyogtató. Funkcionalitásban sem bővelkedik, de a felhasználóknak megad mindent, amit keresnek benne. Ennek viszont előnye is van: az oldal puritánságának köszönhetően az oldalt böngészők figyelme nem oszlik meg hirdetéseken vagy villogó animációkon, hatékonyabbá téve a böngészést. Regisztráció nélkül is lényegében teljes funkcionalitás érhető el. Ennek okából az oldal biztonsága megkérdőjelezhető, viszont mivel nagyon kevés információ tárolódik el a felhasználókról, így a személyes adatok kiszivárgása is minimális.

Az eladókról – ha csak ők maguk nem szolgáltatnak adatokat önszorgalomból – szinte lehetetlen tájékozódni. Mivel nincsenek közösségi értékelések vagy publikus eladói profilok, a platformon nagyon sok a csaló. A helyzetet tovább súlyosbítja, hogy semmilyen érdemi infobiztonsági rendszer nincs implementálva az oldalon.

Azok számára, akik alapos szűrési feltételeket szeretnének használni ez a platform lehetne a legvonzóbb, a kategorizálás is megfelelő, viszont a felhasználói felület elmaradott, statikus és megnehezíti az oldal használatát. Nincsenek korszerű kommunikációs lehetőségek eladó és vevő között.

3.1.4 FACEBOOK MARKETPLACE

Az eladásra szánt termékek eladhatóságát nagyban növeli az, ha lehetőség van tájékozódni az eladóról, legyen szó profilképről vagy felhasználói értékelésről. Ezt a Marketplace platformján oldották meg a legjobban, ahol az eladói profil megtekintése mellett véleményezni is lehet az eladót. Mivel a Facebook Marketplace csak sikeres Facebook regisztráció / bejelentkezés után érhető el, egy alap szintű felhasználói biztonság már adott, mivel interakció csak bejelentkezett Facebook felhasználók között jöhet létre. Maga a regisztráció menete viszonylag egyszerű, de sok adatot kell megadni és nem lehet kizárólag Marketplace profilt létrehozni. Egy Marketplace regisztráció azzal jár, hogy a felhasználó automatikusan a Facebookra is felkerül. Ez sokaknak kellemetlen lehet. Ürömben az öröm, hogy cserébe rengeteg dolog testre szabható a felhasználói profilokon. Mindezek ellenére előfordulnak anonim, támadó jellegű felhasználói értékelések, kifejezetten ilyen célra létrehozott fiókokból, erre egyedül a Marketplace tett érdemi próbálkozást, ahol - a Facebook nagyságának köszönhetően - külön alkalmazottak foglalkoznak a témával és felhasználói bejelentések alapján vizsgálják (és szükség szerint tiltják ki) a potenciális csalókat.



5. ábra – Facebook Marketplace

A saját platformomon a Marketplacehez hasonlóan nagy szerepet fog kapni a közösség ereje. (A támadó jellegű felhasználó értékelések kiszűrésének módszerét egy későbbi pontban fogom részletezni.) Fontosnak tartom azt is, hogy a kapcsolatfelvétel ne merüljön ki a telefonszám és e-mail cím megadásában. Érdemes lehet létrehozni egy chat funkciót, ahol közvetlenebbül és gyorsabban össze lehet kötni a vevőt az eladóval. Erre a Jófogáson és a Marketplacen is láthatunk példát.

Az online apróhirdetési oldalakon a legnagyobb veszélyt nem a hackertámadások jelentik, hanem a hamis hirdetők - attól függően, hogy a hirdető személye, vagy a hirdetendő tárgy az, ami nem fedi a valóságot, így az ezekre való felkészülésre fogom fektetni a hangsúlyt. Mivel ezek olyan tényezők, amik kiszűrésére még nem ismerünk hatékony algoritmusokat vagy gépi megoldásokat, különösen nagy gondot jelent a féken tartásuk.

Ahogy a többi platformnál, itt sincs renden a termékek kategorizálása. Sokszor előfordul, hogy több parcellába is illenének a termékek, néha pedig egyikbe sem. Ez a hiba olyan szinten jelen van, hogy ha az ember egy specifikus dologra szeretne keresni, szinte lehetetlen. A keresések utáni találatok szűrésére lényegében nincs lehetőség. A Facebook Marketplace viszont ebből képes volt előnyt kovácsolni; Mivel a hirdetések reklámként megjelennek a közösségi hírfolyamban is, sokszor olyan termékekről kaphatunk ajánlást, amik felkeltik az érdeklődésünket. Ilyenkor, ha rákattintunk, az átirányít egy olyan oldalra, ahol a többi ajánlást is láthatjuk. Így nyilván nagyobb az esély arra, hogy valamelyik áru elkel. Ez az én platformomon is hasonlóképp fog implementálásra kerülni azzal a különbséggel, hogy nem hírfolyam lesz, hanem egy kezdőoldal, ahol az addigi érdeklődéseink alapján fognak a felhasználók ajánlásokat kapni.

Fontos megemlítenem, hogy a Marketplace technológiai és implementációs oldalról egyáltalán nem hibátlan. A front-end hajlamos a komoly lassulásokra, rengeteg apró, de annál zavaróbb hiba található az alkalmazásban. Az eladásra szánt tárgyak közti keresés nehézkes és pontos találatot szinte lehetetlen kapni a megfelelő paraméterezés hiánya miatt. Kizárólag eladásra van lehetőség: árverések, cserék lebonyolítása nem kivitelezhető.

3.2 BUKOTT PRÓBÁLKOZÁSOK

Természetesen nem én vagyok az első ember, aki hasonló webalkalmazáson gondolkodik, viszont bízok abban, hogy a kivitelezést én fogom a legéletképesebben, a 2021-es elvárásoknak megfelelően megvalósítani. A következő bekezdésekben említést teszek az elmúlt öt év hasonló, sikertelen projektjeiből és igyekszem olyan következtetéseket levonni, amik segítségével én elkerülhetem azokat a hibákat, amikbe mások ezelőtt már beleestek.

3.2.1 BARTAPP – DISCOVERY ROUND

Egy indiegogo.com-os kampányon kívül más nem lelhető fel róla. A koncepciója az, hogy hosszas egyeztetések helyett leegyszerűsítse az kereskedelem folyamatát: ha mindkét felhasználó jobbra húzza a tárgyat, az üzlet végbemegy. Ezen kívül egy olyan rendszert is próbáltak létrehozni, ami tízpercenkénti telefonos megerősítés hiányában sms értesítőt küld egy megadott telefonszámra a felhasználó aktuális helyzetéről. Ennek célja az

üzletkötés folyamatának biztonságosabbá tétele. A bukás okát a funkciók hiányában látom: Véleményem szerint nem nyújtott többet a konkurenciánál, nem lehetett specifikusan megadni, milyen tárgyak lehetnek érdekesek és mik nem.

3.2.2 SWOP.IT

Az alapötlete az, hogy a megunt tárgyakat egy az egyben mások által megunt tárgyakra cserélhetjük. Inkább a jótékonykodáson van a hangsúly, mint az üzleten. Aukciókra és klasszikus eladásokra nincs lehetőség. Egyelőre sem Európában, sem Magyarországon nem terjedt el, de Amerikában vannak államok, ahol használják.

3.2.3 SWAPPY

Egy okostelefonokra szánt alkalmazás, melynek alapjait az adta, hogy a felhasználóknak véletlenszerűen javasolt tárgyakat vételre, majd ezen tárgyak értékelése után a többi javaslat már ahhoz hasonló volt, amik előzőleg pozitív elbírálást kaptak. Lehetőséget kínált arra, hogy az apróhirdetés feltöltése során az eladók ne csak egzakt módon becsülhessék fel értéktárgyaikat, hanem sávosan. A projekt végül nem készült el.

3.3 A VIZSGÁLATOK EREDMÉNYE

Hiszem, hogy a fenti elemzések alapján olyan információkkal és észrevételekkel gazdagodtam, amik segítik szakdolgozatom elkészítését és megálmodott projektem elindulását.

Véleményem szerint webalkalmazásom akkor lehet sikeres, ha kitűnik a tömegből: vegyíti a ma is működő legnagyobb apróhirdetési oldalak erényeit, kijavítja azok hibáit és emellett integrálásra kerülnek benne saját ötleteim. A technológiai megvalósítást jövőállóra és skálázhatóra kell kialakítani úgy, hogy később moduljai cserélhetőek és átalakíthatóak legyenek.

Mindenképp úgy kell megtervezni a személyre szóló hirdetésajánló algoritmust, hogy megfelelő mennyiségű árut ajánljon a felhasználóknak: az a termékajánlatok száma sem túl kevés, sem zavaróan sok nem lehet.

3.4 MOTIVÁCIÓK AZ OLDAL LÁTOGATÁSÁRA

Különösen fontos az, hogy a webalkalmazás akkor is fenntartsa az érdeklődést, ha az oldal látogató épp nem akarnak sem eladni, sem venni. Ezt a következő módszerekkel tervezem elérni.

Az egyik módszer, hogy minden, napi első bejelentkezés után valamilyen bónuszhoz jut a látogató. Ez megnyilvánulhat pontokban, melyek elköltése után kiemelheti a hirdetését vagy kuponokban, amik segítségével olcsóbban juthat hozzá fizetős funkciókhoz. Ennél a rendszernél bevált módszer, hogy minél több egymást követő napon jelentkezik be a felhasználó, annál szignifikánsabb előnyökhöz jut. Manapság az okostelefonokra fejlesztett ingyenes játékoknál, illetve az Aliexpressnél láthatunk példát erre a rendszerre. Amíg előbbinél a célközönség által egy nagyon ellenzett, utóbbinál teljesen elfogadott és közkedvelt módszer. Ennek titka talán abban rejlik, hogy a telefonos játékok szinte megkövetelik a napi bejelentkezést, különben behozhatatlan hátránnyal ijesztgetnek. Ezt a hozzáállást mindenképp kerülendőnek tartom. Ezen módszer akkor működhet jól, ha a felhasználó örömmel, játékként fogja fel a rendszert.

A hírlevelek létrehozása is hasznos lehet, egy sokak által használt klasszikus eszköz. Alapértelmezettként minden regisztrált személy bizonyos rendszerességgel email üzeneteket kap, amikben számukra (remélhetőleg) érdekes cserére ajánlott tárgyakat, eladó tárgyakat, fontos információkat találhatnak. Kihívás lehet megtalálni az ideális e-mail küldési gyakoriságot; el kell kerülni azt, hogy a levelek akár automatikusan, akár az e-mail kliens kezelője által a spam mappába kerüljenek.

A kezdőoldal lesz a webalkalmazás középpontja, így arra kell a legnagyobb figyelmet szentelni. Olyan algoritmusokat kell kidolgozni, amik a bejelentkezett személy keresési előzményei, illetve a mentett hirdetések alapján állítja össze a számára ajánlott tárgyakat.

Az ajánlatoknak jól láthatóan kell megjelenni úgy, hogy a lehető legtöbb kerüljön szemrevételezésre, de ne legyen zavaróan sok. Lehetőséget kell adni egyes találatok későbbi mentésére. A már megtekintett és / vagy mentett apróhirdetéseket meg kell jelölni, így növelhető a felhasználói hatékonyság. Szükséges továbbá egy külön gomb, amivel az olyan tárgyak is megcímkézhetőek, amik az adott személy érdeklődési körén kívülre esnek.

Új felhasználók szerzésének egy módszere lehet az ajánlási rendszer létrehozása is. Ha az egyik - bejelentkezett - „A” személy meglát egy olyan hirdetést, amiről úgy gondolja, hogy egy - a webalkalmazásba még nem regisztrált - „B” ismerősének, kollégájának érdekes lehet, kézenfekvő módot kell biztosítani arra, hogy könnyen elküldhesse neki. Ha „B” végül regisztrál az oldalon és megveszi / elcseréli az értéktárgyat, „A” és „B”-t egyaránt jutalomban kell részesíteni, legyen szó szabadon felhasználható pontokról, kuponokról vagy kiválasztott hirdetéseinek kiemeléséről.

4 SPECIFIKÁCIÓ

Elsődleges célom, hogy más irányból közelítsem meg az online adásvétel témáját, mint a potenciális konkurencia, emellett pedig olyan felhasználói felületet biztosíthassak a felhasználónak, amilyet ők nem tudnak.

A központban a cserekereskedelem áll: számos esetben vonzó lehet az, ha nem pénzért, hanem egy másik, számunkra értékes tárgyért válunk meg értékeinktől. Példának okáért, egy autóeladás: Ha minden nap munkába kell járni vele, sokszor nem kivitelezhető az, hogy miután a felhasználó eladta az autóját, hetekig közlekedési eszköz nélkül maradjon, amíg meg nem találja az újat. Egy ilyen helyzetben sokkal előnyösebb, ha a régi járművek értékegyeztetéssel egy modernebbre lehet váltani. Továbbá sokkal vonzóbb lehet az, ha a megunt tárgyakért azonnal, valami hasznosíthatóhoz lehet jutni. Mindezek indokolják, hogy a konvencionális apróhirdetési oldalaknál több, tárgycseréket önsztönző rendszer legyen implementálva.

A másik fő probléma, amire megoldást szeretnék adni, az a kategorizálás kérdése. Ahhoz, hogy a lehető leghatékonyabban lehessen hirdetésekre szűrni, biztosítani kell egy ajánlási csatornát, aminek segítségével a felhasználók kategorizálási paramétereket küldhetnek be: ha megfelelő mennyiségű kérés érkezik be a rendszerbe, bekerül a felosztási paraméterek közé.

A megfelelő kommunikációt is ki kell építeni a felhasználók között: Nem elég egy telefonszámot megjeleníteni, meg kell teremteni az azonnali üzenetváltás lehetőségét is. Ez később bizonyítékul is szolgálhat, ha valaki csalás áldozata lett, vagy bizonyítania kell, hogy a tárgyak, amikhez a felhasználók hozzájutnak, legális úton cseréltek gazdát.

Arra is gondolni kell, hogy ha az üzlet privát megbeszélés útján ment végbe, akkor is legyen egy dokumentum, ami utólagosan igazolja, hogy minden szabályosan történt. Mivel ilyenkor cserekereskedelem, és nem szokványos eladás történik, adásvételi szerződés nem köthető. Erre a következő megoldást dolgoztam ki: a felhasználói profilok alapján minden végbement üzlet végén automatikusan generálódik egy dokumentum, ami igazolja, hogy hogyan jutottak hozzá a felek a tárgyakhoz. Ebből később egy külön, általánosan használható modult tervezek létrehozni, amit akár külső személyek is használhatnak: piackutatásom során nem találkoztam olyan webalkalmazással, ami lehetőséget nyújt adásvételi szerződések online kitöltésére.

Érdemes lehet létrehozni egy pontozási és jelentési rendszert is, ami jelzi, kik a megbízható eladók és kik azok, akikkel csak nagyobb körültekintéssel szabad üzletet kötni. Ezek klasszikus módon egytől ötig terjedő skálán fognak megjelenni, amik

bejelentkezés után lesznek láthatók. Érdekes lehet továbbá egy ranglista, ahol látható, kik a legmegbízhatóbb és legmegbízhatatlanabb eladók / felhasználók. A legjobb néhány százalékba esőket jutalmakban, a legrosszabbakat pedig megrovásokban lehet majd részesíteni.

Többféle toplista tovább növelheti az oldal látogatottságát. Sokak számára izgalmas lehet egy olyan jegyzék is, ahol a speciálisan nekik összeválogatott tárgyak, vagy épp a legnézettebbek lennének megjelenítve attól függően, hogy kizárólag eladók vagy cserére ajánlották őket. Ezen listákat időrend szerint is kategorizálni kell; ha megjelenik egy olyan tárgy, amit kiemelkedően sokan néztek meg, a fent említett listákon szinte automatikusan elfoglalhatja az első helyet. Ebből kifolyólag pedig a megtekintések már magukat generálják, így károsan befolyásolhatják a listázásokat. Eme problémára megoldást adhat, ha időrend szerint kerülnek kategorizálásra az apróhirdetések. A legtöbb platformon 1-30-100 napos és egy évnyi bontást alkalmaznak.

5 RENDSZERTERV

Az alábbi fejezetekben ismertetésre kerül az elkészítendő szoftver rendszerterve, az irodalomkutatás által szerzett tapasztalatok és következtetések alapján. Bemutatom a létrehozandó adatszerkezet felépítését, kitérve az adatok közti kapcsolatra és relációkra. Ismertetésre kerül a felhasználói felület váza, kulcs funkciói és az azok mögött megbújó logika is.

5.1 ADATOK

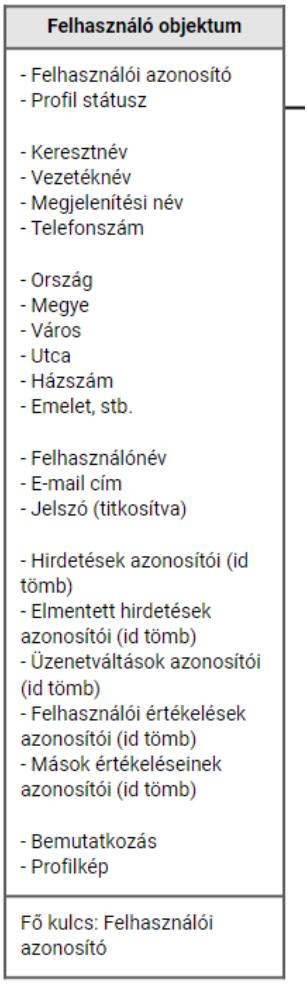
Úgy hiszem, hogy legelőször a szoftver legalsó szintjével, az adatokkal érdemes foglalkozni. Célom, hogy feltérképezzem, milyen információkkal kell majd dolgozzak, illetve kitalálni egy olyan struktúrát, amiben hatékonyan tudom majd őket tárolni. Fontos az is, hogy a rendszerem jól skálázható legyen, így a tervezés során erre külön odafigyelek.

5.1.1 FELHASZNÁLÓK ADATAI

A felhasználók két fő csoportba fognak tartozni, bejelentkezési státuszuk alapján. Mivel sokkal nagyobb forgalom érhető el úgy, ha nem kizárólag bejelentkezett felhasználók használhatnák az oldalt, így a bejelentkezett és vendég felhasználókat külön le kell majd kezelni.

A még nem bejelentkezett felhasználók (továbbiakban vendégek) kizárólag hirdetések keresésére, böngészésére, megnyitására és megosztásukra lesznek korlátozva, így a bejelentkezési státuszukon kívül (igaz-hamis, logikai érték, ezen esetben hamis) mást szükségtelen lesz tárolni róluk a sessionben. Ez nem azt jelenti viszont, hogy nem lesz felhasználói objektum létrehozva. Maga a felhasználó entitás akkor is létrejön, ha nem történik bejelentkezés, egyszerűen csak annyi adatot kap, amennyi az adott helyzetben indokolt.

Egy felhasználó-entitás a következőképp néz ki:



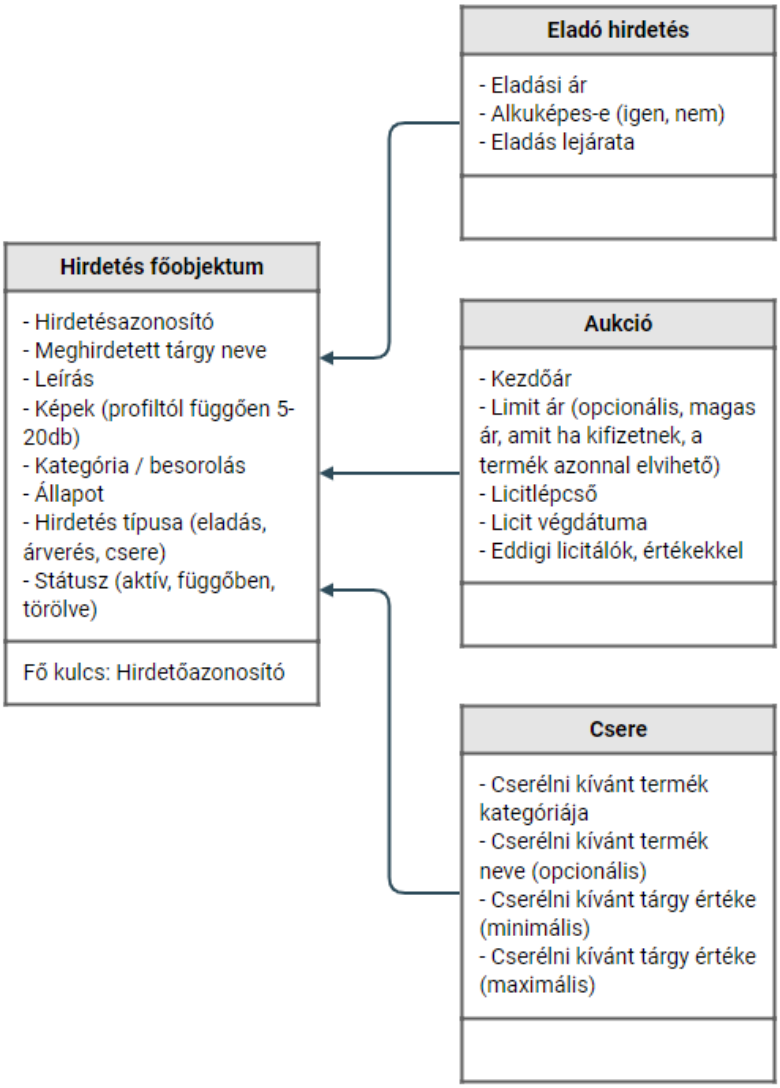
6. ábra – Felhasználó objektum

Mivel a szoftverem úgy tervezem meg, hogy SQL és NoSQL adatbázissal is megfelelően működjön, az adatok relációinak tárolását oly módon fogom megvalósítani, hogy ne függjön az adatmodelltől. Minden dokumentumnak lesz ID-je, ezen fájlok belsejében hivatkozások lesznek a velük kapcsolatban álló dokumentumok ID-jeire.

5.1.2 HIRDETÉSEK ADATAI

Az apróhirdetések, cserére ajánlott tárgyak számos adat rögzítését kívánják meg. Mivel alapvetően háromféle (eladó, licitre bocsátott, csere) hirdetést különböztetünk meg, érdemes lehetne háromféle hirdetés-objektumot létrehozni. Ez esetünkben nem működhet, mert nem csak külön-külön fordulatnak elő ezen objektumok, hanem egymás permutációjaként is képezhetnek egy entitást (Miért ne lehetne egy tárgy cserére bocsátva úgy, hogy közben eladó is?). A hirdetések adatstruktúráját úgy fogom felépíteni, hogy van egy alap objektum, néhány esszenciális adattal, amihez ha szükséges, további kiegészítő objektumokat csatolok és így alkotnak egy egészet. Így mindig csak annyi

adatot fognak tartalmazni, amennyi feltétlen indokolt és nem lesz feleslegesen sok üres mező sem.



7. ábra – Hirdetés objektum

5.1.3 LOGISZTIKAI ADATOK

A felhasználók és apróhirdetések adatain kívül számos mező eltárolására is szükség van a platform megfelelő működéséhez. El kell tárolni a megnyitott oldalt, a sütiket és ahhoz, hogy megfelelően funkcionáljon a hirdetésajánló funkció, a felhasználó által kedvelt (amikre cserét ajánlott) hirdetések előzményeit is. Az ezeket és ezekhez hasonló adatokat (továbbiakban logisztikai adatok) szintén szükséges kezelni, melyekre néhány példa:

- Megnyitott oldal
- Sütik
- Megjelenítendő apróhirdetések, ajánlások - ehhez oldalszám
- Felhasználó adatai
- Üzenetek
- Értesítések – outbideltek[2], megnyerted, lejárt, stb.
- Pontos idő - ehhez mérten kell majd
- Hirdetéskereső adatai
- További információk: mennyi ideje nézik a hirdetést stb.

5.1.4 ADATKAPCSOLATOK

Az alapadatok összeszedése és csoportosítása után folytassuk a köztük lévő kapcsolatok felállításával. A platform és ezzel együtt a felhasználók biztonsága érdekében fontos, hogy minden felhasználó csak annyi adathoz férjen hozzá, -és csak annyit rögzítsünk róla- amennyi feltétlenül szükséges. Az adatkapcsolatokat is ennek megfelelően kell felállítani.

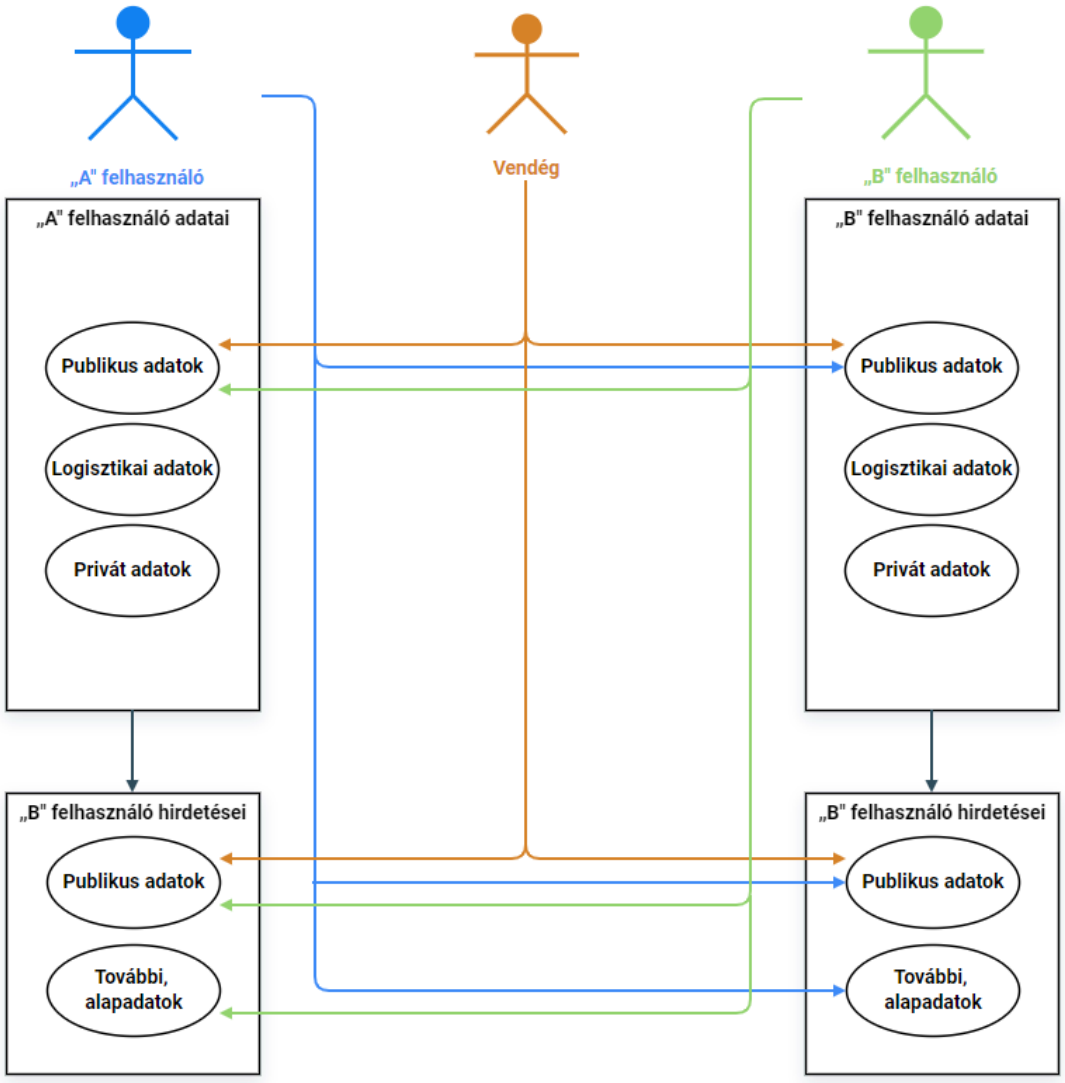
Egy azonosítatlan, vendégfelhasználó hozzáférhet az összes létező aktív hirdetéshez, kereshet rájuk és szűkítheti a találatokat. Láthatja a hirdető nevét, profilképét, bemutatkozását, e-mail címét és telefonszámát. Utóbbi alapértelmezetten rejtve van, így a különböző telefonszámokat gyűjtő káros programok nehezebben férnek hozzá. Semmilyen módosítási joga nincs.

A felhasználó hozzáférhet B felhasználó: e-mail címéhez, telefonszámához, meghirdetett termékeihez (eladás, aukció, csere), értékeléseihez, bemutatkozásához, chat[3]elhet vele. Módosítási joga egyikhez sincs.

„A” felhasználó hozzáférhet a saját: e-mail címéhez, telefonszámához, meghirdetett termékeihez (eladás, aukció, csere), jegelt hirdetéseihez, eladott hirdetéseihez (korlátolt adat), eladott termékeihez, felhasználónevéhez, módosíthatja saját jelszavát, törölheti hirdetéseket, változtathat a telefonszámán, e-mail címén, visszavonhatja aukcióit (ha még nem érkezett rájuk licit), használhatja a chatet, törölheti fiókját

„A” felhasználó hozzáférhet, „B” felhasználó hirdetéséhez: láthatja a képeket, leírást, eladási árat / mire cserélne /, a termék földrajzi helyét, elmentheti későbbre, láthatja a hirdető nevét, elérhetőségeit (név, e-mail, telefonszám, chat)

[2] Valaki nagyobb összegben licitált a tárgyra, így már ő vezeti az árverést
[3] Élő írásbeli kommunikáció



8. ábra – Felhasználói jogkörök

5.2 FOLYAMATOK, FUNKCIÓK

Az adatok közti kapcsolatok megismerése után a platformon lezajló főbb folyamatok ismertetése következik.

5.2.1 REGISZTRÁCIÓ

Az új felhasználói fiókok létrehozása kulcs funkció, a következőképp kell lefolynia: Miután a felhasználó megadta a feltétlen szükséges adatokat (e-mail cím, jelszó, megjelenítési név), lehetősége nyílik további információkat rögzíteni, melyek az oldal további használata során indokoltak lehetnek (vezetéknév, keresztnév, telefonszám, ország, megye, város, irányítószám, utcanév, házszám, bemutatkozás). Ezután a

regisztráció elindul, a megadott adatok ellenőrzése után rögzítésre kerülnek, létrejön a felhasználói azonosító (e-mail cím) és alapértelmezett állapotba kerülnek a további adatok, pl. meghirdetett termékek, üzenetek stb.

A folyamat végén a látogató üzenetet kap annak sikerességéről, majd elérhetővé válik a bejelentkezés.

5.2.2 BEJELENTKEZÉS

A bejelentkezés elindításához két mezőt kell kitölteni: e-mail cím és jelszó. Ha ez megtörtént, a két adat ellenőrzésre kerül: ha van találat az adatbázis oldaláról, létrejön a login-token[4] és a vendég bebocsátást nyer a kezdőoldalra, ahonnan pedig már felhasználóként hivatkozhatunk rá. A tokent kizárólag kliens oldalon tároljuk a munkamenetben, 30 perces inaktivitás után törlődik. Abban az esetben viszont, ha a felhasználó úgy zárja be a böngészőt, hogy a munkamenetből a kijelentkezés nem történt meg (nem jelentkezett ki, vagy nem volt inaktív), az alkalmazás legközelebbi megnyitásakor a bejelentkezési állapot megmarad.

5.2.3 FELHASZNÁLÓI FIÓK TÖRLÉSE

A felhasználói fiók deaktiválása kétféleképp érhető el. Kezdeményezhető a bejelentkezett látogató felől, jelszavas megerősítés után. Ez a folyamat visszafordítható, a fiók adatai megmaradnak, viszont inaktiválásra kerül. A másik lehetőség a teljes törlés. Ezt az adminisztrátorok indítják és rendszerint akkor történik meg, ha a hirdető megszegi a felhasználási feltételeket, vagy ha a fiók legalább egy éve nem jelentkezett már be. Ilyenkor minden, a törlendő fiókhoz tartozó adat megsemmisül és igény esetén az e-mail cím tiltólistára kerül.

5.2.4 HIRDETÉSEK KERESÉSE

Az apróhirdetések keresését többféle módon lehet paraméterezni.

Népszerűség szerint: Ha így keresünk, be lehet állítani, milyen idősávra (adott nap / hét / hónap / év) és milyen szempontból (legtöbbet megtekintett, legtöbb üzenetet kapott, legtöbbet kívánságlistára adva) keresünk.

Konkrét név alapján: Természetesen ha már megvan, hogy a látogató mit keres, egyszerűen be is gépelheti a tárgy megnevezését.

[4] A bejelentkezések tokenekkel – zsetonok, melyek hitelesítő kulcsokat tartalmaznak – valósulnak meg

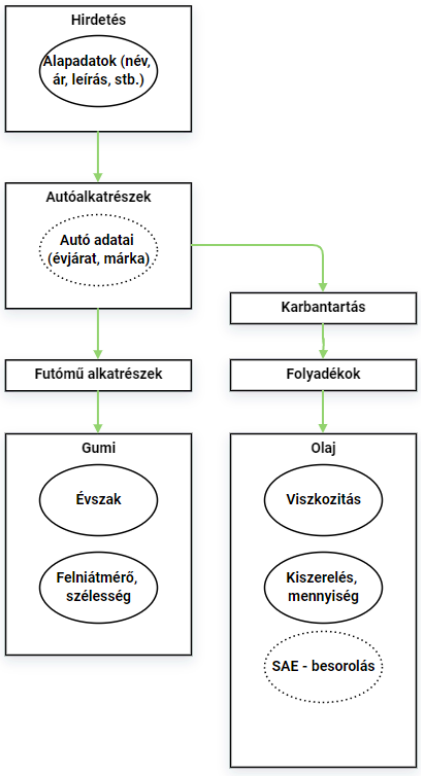
Kategorizálás alapján: A megfelelő kategória és ársáv kiválasztása sokszor kellően leszűkíti az ajánlatokat.

Apróhirdetés típusa alapján: A termékek listázása aszerint is szűkíthető, hogy eladásra, aukcióra vagy cserére vannak-e bocsátva.

A keresés elindítása után egy SQL lekérés fut végig az adatbázison, majd az egy json dokumentum formájában választ küld vissza a front-endnek.

5.2.5 HIRDETÉSFELADÁS

A hirdetésfeladás menetét nagyban befolyásolja az, hogy milyen kategóriájú terméket kívánunk feltölteni. Amit kötelezően meg kell adni, az a tárgy neve, állapota, leírása, becsült értéke (erre azért van szükség, hogy ezekre alapozva el lehessen indítani a csereajánlatokat), minimum egy kép. Besorolástól függően további mezők jelennek meg, amiknek egy része kötelező, egy része opcionális.



9. ábra - Kategorizálás

Az ábra alapján egy 5 literes kiszerelésű, személyautóba való motorolaj meghirdetésénél a következő adatokat lehet megadni: Név, ár, leírás Autó évjárata (opcionális, de bizonyos márkáknál hasznos lehet, hiszen egy 1982-es 1.3 Ford Escortba nem

ugyanolyan olaj való, mint egy 2008-asba), márkája (opcionális) Olaj viszkozitása, kiszerelése/mennyisége, SAE-besorolása (opcionális).

5.2.6 CSEREAJÁNLAT KÜLDÉSE

Ha egy adott termék úgy meg meghirdetve, akkor más felhasználók cseréket ajánlhatnak az aktuálisan megtekintett tárgyakra. Ennek során a cserét ajánló felhasználó elküldi a felhívást, majd később kiválasztja, hogy mit/miket ajánlana a tárgyért cserébe. Lehetőség van arra is, hogy ráfizessen, vagy ráfizetést kérjen a másik tulajdonostól, amit a másik tulajdonos elfogadhat vagy elutasíthat. A további alkudozás sincs kizárva.

5.2.7 ADÁSVÉTELI SZERZŐDÉS GENERÁLÁSA

Amikor a két fél között megegyezés születik és az üzlet végbemegy, automatikusan legenerálódik egy PDF formátumú adásvételi szerződés a vevő és eladó adataiból, az ők előzetes beleegyezésükkel. Ez opcionális, de ha az egyik tulajdonos ragaszkodik hozzá, a másiknak el kell fogadnia, vagy nem megy végbe az eladási/cserefolyamat. Mindkét félnek joga van elállni az üzlettől, ha a szerződést tartalmilag vagy formailag nem találja megfelelőnek. A dokumentum tartalmazza a nevüket, címüket, az értéktárgy adatait, a platform hitelesítő kódját, dátumot és szabadon kiegészíthető további információkkal vagy egyéb záradékokkal.

5.2.8 CHAT FUNKCIÓ

Egy megfelelően funkcionáló kommunikációs csatorna nagyban megkönnyítheti az adott termék eladását. Ennek apropójára elengedhetetlennek tartom azt, hogy platformomon az emberek ne csak telefonon vagy e-mailben egyezhessenek meg egymás közti üzleti folyamataikról, hanem valós időben, szövegesen is. Ennek során élő üzeneteket válthatnak egymással, akárcsak a Facebook Marketplacen vagy a Jófogáson.

5.3 FEJLESZTÉSI TERV

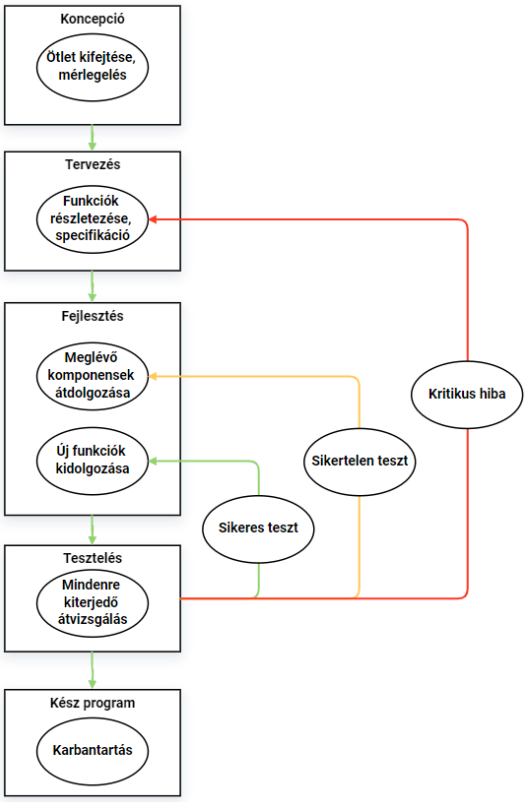
Igyekszem minden olyan problémát ismertetni, ami szakdolgozatom kidolgozása és fejlesztése során tudomásomra jutott. Először is szükség lesz egy adatbázisra, a tényleges programozói munka során ennek létrehozásának és felállításának kell az első lépésnek lennie. Miután ez megtörtént, az üzleti logika kialakításán lesz a hangsúly.

Ennek során ügyelni kell arra, hogy a függvények és funkciók kellően szét legyenek választva és ennek ellenére ne jelenjen meg redundancia. Az entitásoknak az objektum orientált programozás elvei szerint csak az abszolút szükséges rálátásuk egy módosítási lehetőségük legyen egymásra úgy, ezzel kiszűrve a monolitikus, „mindent-csináló”

komponensek megjelenését. A CRUD [5]metódusok kialakítása után a NON-CRUD függvények következnek, amiknél külön figyelmet fordítok a megfelelő skálázhatóságra.

Saját magam által futtatott adatbázis esetén ezeket Postmannal fogom tesztelni. Ha a tesztelés sikeresen lezárult, elkezdem a webalkalmazás megjelenítésének (front-end) kiépítését, ami a back-enddel párhuzamosan fog haladni.

A teljes folyamat során kellő figyelmet fogok fordítani a tesztelésre. Érdemes külön teszteseteket írni azon helyzetekre, amik kiszűrik a felhasználói hibákból adódó hibás működést is - ahogy az a valós életben is várható - és erre fel is készítem a program különböző komponenseit.



10. ábra – Fejlesztés menete

Sikeres projekzárást és megfelelően működő alkalmazást csak folyamatos teszteléssel fogok tudni elérni, így különös figyelmet kell fordítanom arra, hogy a különböző hibákat már írmagjukban elfojthassam, ehhez pedig az kell, hogy minél hamarabb felderítsem őket. Először az alap programkeretet hozom létre, majd egyesével lefejlesztem hozzá a funkciókat, fontossági sorrendben. Ha ezek tesztelése során hiba lép fel, javítom. Ha a tesztek sikeresen lefutnak, elkezdem a soron következő modul implementálását. Ha

[5] Create – Read - Update - Delete

ellenőrzésem során kritikus hiba lépne fel, elképzelhető, hogy egy réteggel visszább kell lépnem és módosítanom kell a specifikáción. Erre példa lehet az, ha kiderül, hogy az általam megvalósítani kívánt funkció a gyakorlatban nem megvalósítható, mert a többi, esszenciális programrész kizárja annak lehetőségét, hogy akaratom szerint működjön. Ez mindenképp kerülendő, de erre is fel kell készülni.

6 TECHNOLÓGIA VÁLASZTÁS ÉS INDOKLÁS

Ezen fejezetben az alapproblémák bemutatása után két csoportra (back-end és front-end) fogom választani az általam választott, fejlesztés során használt technológiákat. A nagyobb problémákhoz igyekszem bemutatni más, az általam választott eszközöktől eltérő lehetőségeket és megindokolni, hogy miért nem rájuk esett a választásom.

6.1 ADATTÁROLÁS

Programom komplexitása miatt számos fontos technológiai döntést megkövetel. Alulról elindulva az egyik az, hogy milyen adatbázist használjak. Kétféle lehetőségem van, SQL, azaz relációs, illetve NoSQL, azaz nemrelációs adatbázisok.

Utóbbi előnye az, hogy gyorsan változó, strukturálatlan adatok nagy mennyiségének kezelésére is képesek. Akkor érdemes használni, big-data jellegű, óriási változatosságú, látszólag egymással kapcsolatban nem lévő információt kell eltárolni és keresni. Bizonyos módosításokkal és fejlesztésekkel viszont relációs adatbázis is szimulálható vele, aminek segítségével relációs adatbázisokhoz hasonló működés érhető el.

Előbbi, az SQL adatbázisok előnye pedig ezzel szemben az, hogy rendszerezett, egymással szigorú kapcsolatban álló, de óriási mennyiségű adatok kezelését sokkal egyszerűbb módon képes végezni már kevés fejlesztéssel is. Ha az információ fix struktúrában tárolható, érdemesebb ezt választani. Hátrányuk az, hogy ha menet közben változtatnánk az adatok struktúráján, az komoly utómunkával járna és ilyenkor a migráció is körülményes. Habár ez egy jól kiforrott, teljesen profi módon megtervezett alkalmazásnál egyáltalán nem valószínű, én viszont meg szeretném hagyni magamnak ezen lehetőséget.

Mivel alkalmazásomnál prioritás az elérhető legnagyobb techstack-függetlenség, úgy fogom létrehozni azt, hogy SQL és NoSQL adatbázisokra is fel lehessen húzni azt.

A jelenleg elérhető nyílt forráskódú adatbázisok közül a PostgreSQL és a MySQL a legnépszerűbb, előbbinél egyszerűbb a skálázhatóság és natív megoldást kínál NoSQL konverzióra is, így ezt választottam.

Emellé szerettem volna egy tisztán felhőalapú adatbázist is, ezekből a Firebase és az Amazon AWS a legelterjedtebb. Mindkettővel próbálkoztam, viszont az AWS üzleti modellje kevésbé megengedő, már az első hónapban 10 dolláros számlát állítottak ki, amíg a Google megoldása nagyobb adatmennyiség tárolásánál és feldolgozásánál is ingyenes maradt, így nem is volt kérdés, hogy utóbbit választom a szakdolgozatomhoz.

6.2 BACK-END

Szoftverem szerveroldali komponenseinek stabilan működőnek és jól skálázhatónak kell lenniük. A programozási nyelv választásakor az elsődleges szempont a lehető legteljesebb funkcionalitás elérése volt, attól függetlenül, hogy melyik programozási nyelvben van nagyobb gyakorlatom. Ezeknek függvényében két nyelvre, a Javascriptre és a C#-ra szükítettem le keresésem.

A Javascript (esetemben Node.JS, Next.JS) előnye az, hogy nagyon gyors fejlesztést tesz lehetővé, nagy szabadsággal, minimalista, tanulása könnyű és nem csak back-endhez, hanem könyvtárai segítségével front-end oldali feladatok megoldására is tökéletesen használható. Ezen tulajdonságát ki is használom a továbbiakban. Mivel Open-Source nyelv, rengeteg információ és tananyag áll rendelkezésre elsajátításához. Hátránya, nem konzisztens és a kódolás, illetve program futása során előforduló hibák kiszűrése (debugolás) kifejezetten nehézkes.

A C# ezzel szemben egy sokkal kötöttebb nyelv. Rendkívül stabil, a fejlesztés a Visual Studio segítségével pedig majdnem ugyanolyan egyszerű, mint Javascriptben. A hibafeltárás általában gyorsabban működik, sokszor már a program futtatása előtt lehet látni, hol vannak a kritikus programrészletek. A pedig szintaxisa – a nyelv szigorúságából kifolyólag – magától értetődő. Manapság sokkal kevésbé népszerű, mint a Javascript, ami bizonyos szempontokból hátrányos lehet.[2]

Szóba jöhet még a React Native is, mint backend, ami ugye szintén Javascript alapú, de az sokat rontana a platformfüggetlenségen, így nem élek vele.

6.3 FRONT-END

Ha Front-Endről van szó, alapvetően rengeteg választási lehetőség áll rendelkezésre. Kezdve az Angulartól, a Flutteren és a Reacton - és még sok máson - át a Vue-ig bárki megtalálhatja a számára megfelelő fejlesztéshez szükséges eszközöket.[3]

Az Angular – mivel nyelve a Typescript – alapvetően egy olyan keretrendszer, amiben fejleszteni viszonylag egyszerű és gyors. A React ezzel szemben csak egy könyvtár, amit eredetileg csak nézetek létrehozására szolgált, de a Flux segítségével mindenre képes lehet, amire az Angular és a Vue is.

	Angular	React	Vue
Fejlesztés nehézsége	Nehéz, átfogó tudást igényel Typescriptben és MVC-ben is.	Könnyen elkezdhető, de haramdik féltől származó könyvtárak használata is szükséges	Magas testreszabhatóságot kínál, ennél fogva a tanulása is könnyebb, mint a Reactnak vagy az Angularnak
Kiforrottság	A legkiforrottabb, abszolút elterjedt	Számos támogató, sok munkahelyen használják, startupok is	A legújabb a három közül, az utóbbi néhány években lett felkapottabb
Frissítések	6 havonta	Stabilizáló javítások szinte azonnal, nagyobb frissítések 8-10 havonta	7-12 havonta
Népszerűség	Népszerű	Legnépszerűbb	Még nincs elterjedve
Teljesítmény	Jó	A három közül a legjobb	Jó, de nem éri el az Angular szintjét

1. táblázat – Angular-React-Vue összehasonlítás

A jelenleg népszerű és viszonylag egyszerűen használható front-end technológiák közül a Reactot választottam a következő okokból: Mivel a Facebook hozta létre, megfelelő pénzügyi és szakmai támogatással rendelkezik, dokumentációja kiváló és kellő fejlett ahhoz, hogy minden elképzelésem megvalósíthassam benne. További nagy előnye, hogy React-tel XML, HTML mezőket lehet a JavaScript fileokba írni JSX szintaxissal, ami nagyon felgyorsítja a fejlesztés folyamatát. ES6 szabványt és vanilla JavaScript nyelvet is támogat, így kellően flexibilis is. Emellett még szóba jött az a Flutter is, de korábbi tapasztalataim alapján a React segítségével ugyanolyan hatékonyan megvalósíthatom elképzeléseimet, ha mellé Next.JS-t is használok.

7 FEJLESZTÉS MENETE

Miután a fentiekben eldőlt, hogy milyen technológiákat fogok használni a fejlesztés során, ezen fejezetben dokumentálom a folyamatot, külön kitérve az általam fontosnak tartott problémák és azon megoldásainak bemutatására.

7.1 FEJLESZTŐI KÖRNYEZET, KIEGÉSZÍTŐK

A programírás elsődleges eszköze a Microsoft Visual Studio Code-nak az szakdolgozatom írásakor elérhető legújabb stabil kiadása lesz, ami a v1.73-as kódszámot viseli. Az egyszerűség kedvéért a jegyzeteimet is ebben írtam. A fejlesztés fő nyelvének Javascriptet választottam, Reacttal, Next.JS-sel és helyenként Typescripttel, így ki tudtam használni az NPM/NPX[6] modulok hozzáadott funkcionalitásait is, amiktől C# esetén elestem volna.

Számos VS-Code kiegészítőt is használok, ezek közül a legfontosabbak:

- AL Lint[7]: Monitorozza a megírt kódot és jelez, ha valamit egyszerűsíteni lehet.
- Debugger for Chrome: Kibővíti a böngészőből elérhető hibakeresési eszköztárat.
- ESLint, JSX Snippets, ES7+ and React snippets: Code snippeteket[8] biztosítanak, melyekkel néhány karakter leírásával előhívhatóak lesznek komplett programvázak, amiket már csak meg kell tölteni funkciókkal.
- Prettier: Kódformázó, segítségével uniform programkód hozható létre anélkül, hogy túl sok energiát kellene belefektetni.
- Tailwind CSS Intellisense: Tippekkel, magyarázatokkal segíti a Tailwind[9] használatát.

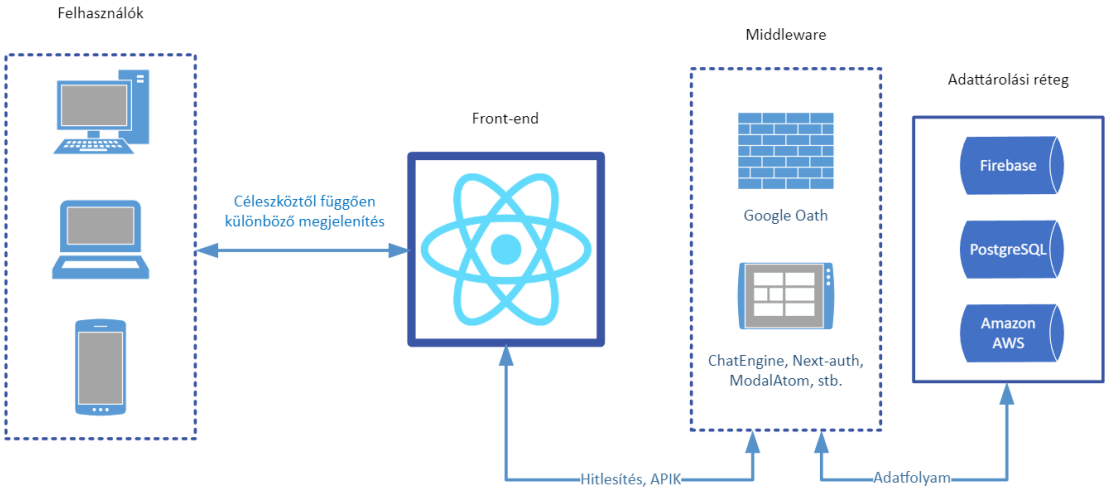
A verziókezeléshez a 8.10.3-as verziójú GitKraken Pro-t használom, mert eddigi fejlesztői munkám sokán gyakran bizonyultak hasznosak azok a plusz funkciói, amit egy egyszerűbb (például SourceTree) vizuális git klienssel szemben nyújtani tud. Természetesen a branchjeim nem csak localban[10] vannak meg, hanem Githubon is, ahogy a szakdolgozat azt ki is köti. A szenzitív adatokat .env fájlokban tároltam, így nem állt fent a veszélye annak, hogy egy óvatlan git push során kikerülnek az internetre. Magát a fejlesztést Windows 11-en és MacOS Monterey-en végeztem, attól függően, hogy épp melyik eszközöm volt kéznél. Ezen módszer segítségével, a teljes programírási folyamat

[6] Node Package Manager – csomagkezelő Node.js-hez
[7] VS Code bővítmény, elemzi a kódot és segít az egyszerűsítésében
[8] Kódrészletek, melyek komplett kódrészekké alakulnak
[9] A legnépszerűbb CSS-keretrendszer segédprogam. használatával kevesebb CSS classnevet kell használni.
[10] Kizárólag a saját számítógépemen

alatt bizonyos lehettem arról, hogy az alkalmazásom platformfüggetlen marad. API[11]-k paraméterezéséhez és teszteléséhez Postman-t használtam, szintén az elérhető legfrissebb, v10.3.5-ös verzióval.

7.2 MOBILALKALMAZÁS, PLATFORMFÜGGETLENSÉG

Próbáltam több módon is biztosítani azt, hogy az alkalmazásom a lehető legtöbb platformon elérhető és használható lehessen függetlenül a perifériáktól, a kijelző méretétől, felbontásától és az operációs rendszertől. Ennek megvalósítása felé az első lépés az volt, hogy – mivel két különböző operációs rendszeren végeztem a fejlesztést – a teljes programírási és tesztelési folyamat alatt felváltva láthattam az eredményt különböző eszközökön.



11. ábra – A rendszer felépítése

A fenti architekturális ábrán jól látható, hogyan különülnek el a rendszerem különböző alkotóelemei. A legalsó szint természetesen az adattárolási / Back-end réteg, itt nem egy specifikus és kötött alkotóelemet ábrázolok, hanem többet, amik szabadon felcserélhetőek attól függően, hogy épp a körülmények figyelembevételével mi a legoptimálisabb. Középen a Front-end helyezkedik el, itt valamennyivel másabb a helyzet. Bár kísérleteztem többféle megjelenítési réteggel, egyelőre a React+Next.JS verziót tartom az igényeimnek legmegfelelőbbnek, de itt sincs kizárva a módosítás lehetősége. A Middleware-réteg olyan köztes szoftvereket tartalmaz magában, amik megkönnyítik a fejlesztés, vagy afféle profi megoldásokat nyújtanak specifikus problémák kezelésére (Például komoly titkosítással rendelkező autentikációs szolgáltatás vagy külső könyvtárak, amik segítik az oldalon belüli navigációt, esetleg megkönnyítik a tartalom megjelenítését), amiket saját kezűleg lefejleszteni túl sok időt / munkát jelentene.

[11] Application Programming Interface – hozzáférés biztosít egy adott szoftver utasításkészletéhez

A felhasználói réteg tartalmazza a rendszer összes potenciális használóját, nekik közvetlen kapcsolatuk csak a megjelenítési réteggel van.

7.3 ADATBÁZIS, BACK-END

Mivel alkalmazásomat jövőállóra szeretném elkészíteni, SQL és NoSQL adatbázisokkal egyaránt működnie kell. Ezt oly módon biztosítottam, hogy mindkettővel készítettem prototípust.

Először egy klasszikus, localban létrehozott és futtatott SQL (PostgreSQL) adatbázist hoztam létre ahol a Back-end logikát a nulláról kellett felhúzni. Az adatok vizuális megtekintéséhez a PostgreSQL által ajánlott pgAdmint használtam. Ennél a megvalósításnál a CRUD metódusokat egyszerű SQL query[12]-k segítségével építettem fel:

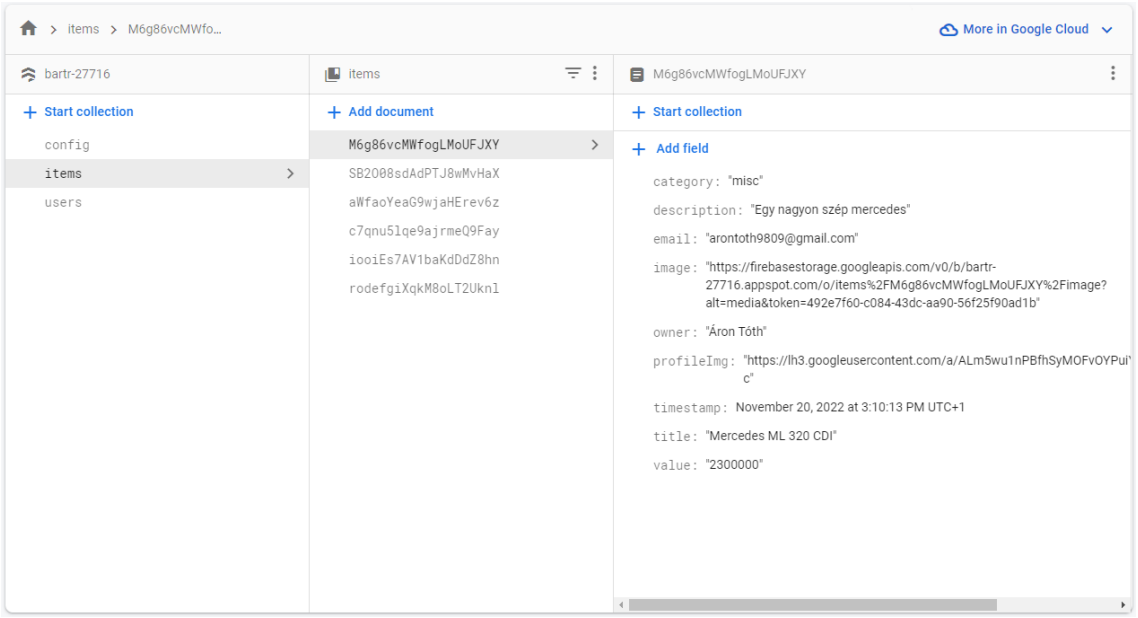
```
//update
app.put("/items/:id", async (req, res) => {
  try {
    const { id } = req.params;
    const { description } = req.body;
    const updateItem = await pool.query(
      "UPDATE items SET description = $1 WHERE item_id = $2",
      [description, id]
    );

    res.json("Item is updated!");
  } catch (err) {
    console.error(err.message);
  }
});
//delete
app.delete("/items/:id", async (req, res) => {
  try {
    const { id } = req.params;
    const deleteItem = await pool.query(
      "DELETE FROM items WHERE item_id = $1",
      [id]
    );
    res.json("Item is deleted!");
  } catch (err) {
    console.error(err.message);
  }
});
```

12. ábra – Apróhirdetés módosítása és törlése SQL Back-enddel.

Ez követte a NoSQL megvalósítás, amit bár nagyobb kihívásnak éltem meg, úgy hiszem, jelen helyzetben jobban illik az elképzeléseimhez, így erre helyeztem nagyobb hangsúlyt. Itt először AMAZON AWS-sel próbálkoztam, de üzleti modelljük miatt hamar áttértem a Google Firebase-re. Utóbbi jó döntésnek bizonyult, ugyanis sok olyan funkcióját

[12] Lekérdezés

használom (mint például a saját adatok felhőből való egyszerű módosítása, valós idejű szinkronizálással), ami – bár helyettesíthető saját és harmadik féltől származó megoldásokkal, illetve úgy oldottam meg, hogy ne támaszkodjak rájuk túlságosan – hasznomra vált. A fő adatbázis létrehozása egyszerű, ugyanis a Firebase saját felületén is lehetséges – és mivel NoSQL, nagyon kevés korlátozó tényező van -, de emellett kódból is megoldható.



13. ábra – Adatok tárolása a NoSQL felhőadatbázisban.

Amit érdemesnek tartok megemlíteni, hogy az Amazon AWS vagy a Google Firebase (esetleg Microsoft Azure, stb.) és a felhő-hosztolás [13]segítségével könnyen biztosítható a Back-end létező legmagasabb rendelkezésre állása, amíg egy saját infrastruktúrával rendkívül körülményes és költséges lenne.

A felhasználói autentikációt úgy terveztem meg, hogy alapértelmezett módon saját Google fiókkal lehessen bejelentkezni, így nem csak a regisztrációs és bejelentkezési folyamat egyszerűsödött le, hanem a felhasználói fiókok védelmét is növelhettem olyan eszközökkel, mint a beépített kétlépcsős azonosítás vagy a túlterheléses támadások egy része ellen védő CAPTCHA[14] védelem. Ilyen módon az is biztosított, hogy ha a felhasználó elfelejti a jelszavát, újat kérhessen.

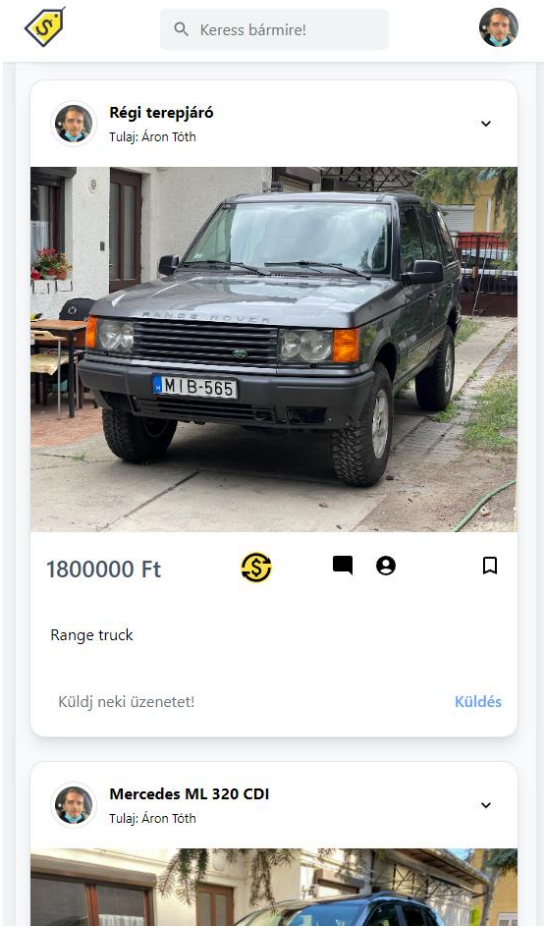
[13] Az alkalmazás / adatszolgáltatás nem a saját gépemen, hanem a felhőből történik
[14] Olyan feladvány, aminek megoldása segítségével kiszűrhetőek az igazi és bot felhasználók

7.4 KLIENS, FRONT-END

A Front-endnél a platformfüggetlenség mellett fontos szempont volt az, hogy a mai elvárásoknak megfelelően átlátható, letisztult, de reszponzív legyen. Ennek eléréséhez az alap React-Next.JS-Typescript kliensemhez nagy mennyiségben telepítettem külső NPM modulokat. Amit mindenképp megemlítenék az a Tailwind, a Material-UI, a Next-Auth, a Recoil és a React-Tinder-Card. Ezzel a megoldással azt is elkerülhettem, hogy az oldalt manuálisan frissítgetni kelljen, mert ott volt a Virtual Dom[15].

A felhasználói felület összerakásánál arra törekedtem, hogy bármilyen eszközön is legyen megjelenítve a webapp, az használható és intuitív maradjon, sőt, amennyiben lehetséges, ki is szerettem volna használni az épp aktuális platform előnyeit.

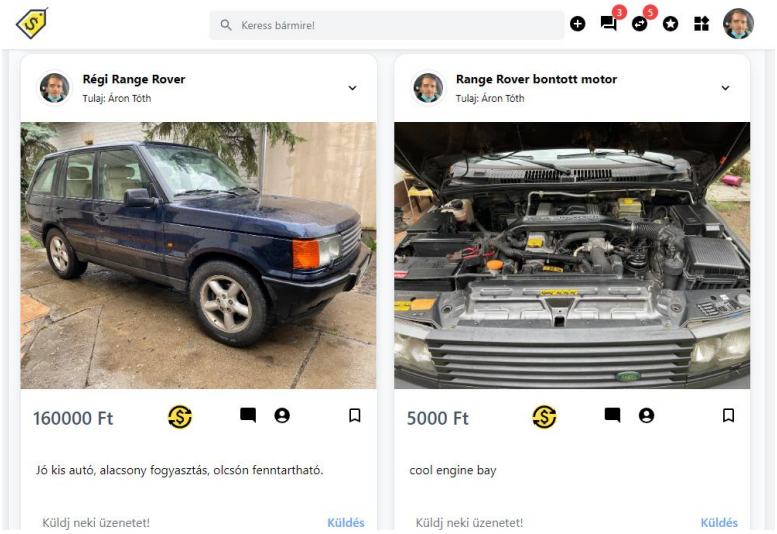
Alapvetően három fő eszköztípust tartottam szem előtt a fejlesztésem során: PC, Tablet és Mobil.



14. ábra – Mobil-nézet

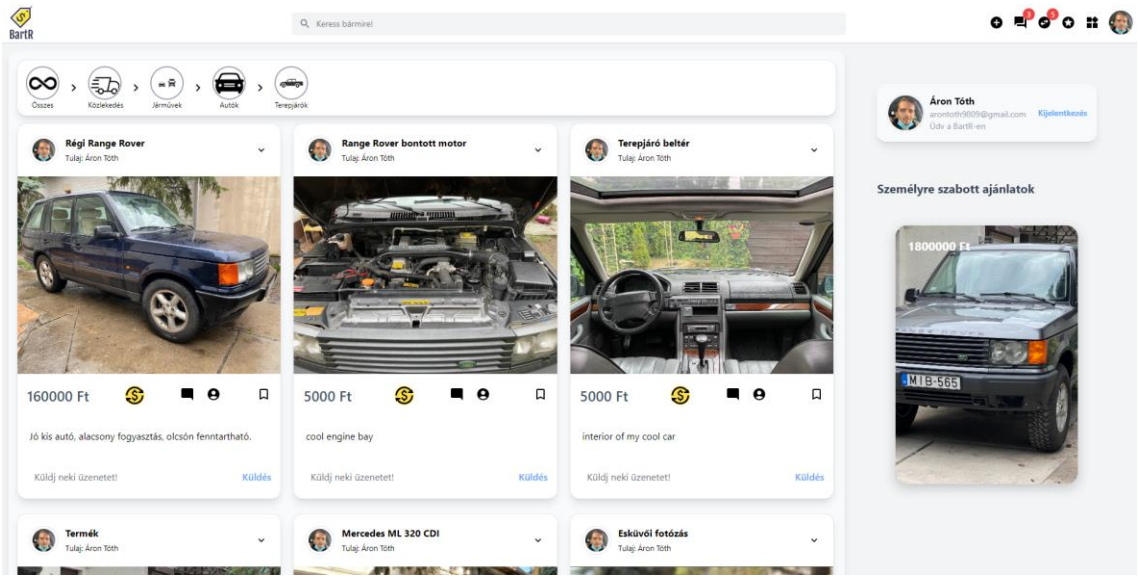
[15] Az alkalmazás felhasználói felületét jelenti, viszont új, gyorsabb technológiával

A mobilos felület úgy lett konfigurálva, hogy - mivel általában csak egy kis képernyő adott - ne jelenítsen meg túl sok, potenciálisan zavaró információt egyszerre, hiszen az inkább csak elvenné a felhasználó kedvét a böngészéstől. Ebben a nézetben bizonyos összetevők (pl. a termék becsült értéke, vagy az üzenetküldési panel) nagyobb méretben kerülnek megjelenítésre. A hirdetések egy oszlopban, kártyákként jelennek meg, a többi funkció pedig a főképernyőn kívüli panelekről érhető el.



15. ábra – Táblagép-nézet

Táblagép-nézetben a hirdetések már két oszlopban érhetők el, ha viszont valakinek mégis az egy oszlopos megjelenítés tetszene jobban, a képernyő elforgatásával egy nagyított mobilos nézet csalogatható elő. A jobb felső sarokban (amennyiben elég magas a kijelző felbontása és elférnek) már megjelenhetnek a főbb vezérlő ikonok (Új hirdetés létrehozása, chat-funkció, csereajánlatok, elmentett hirdetések, összes hirdetés) is.



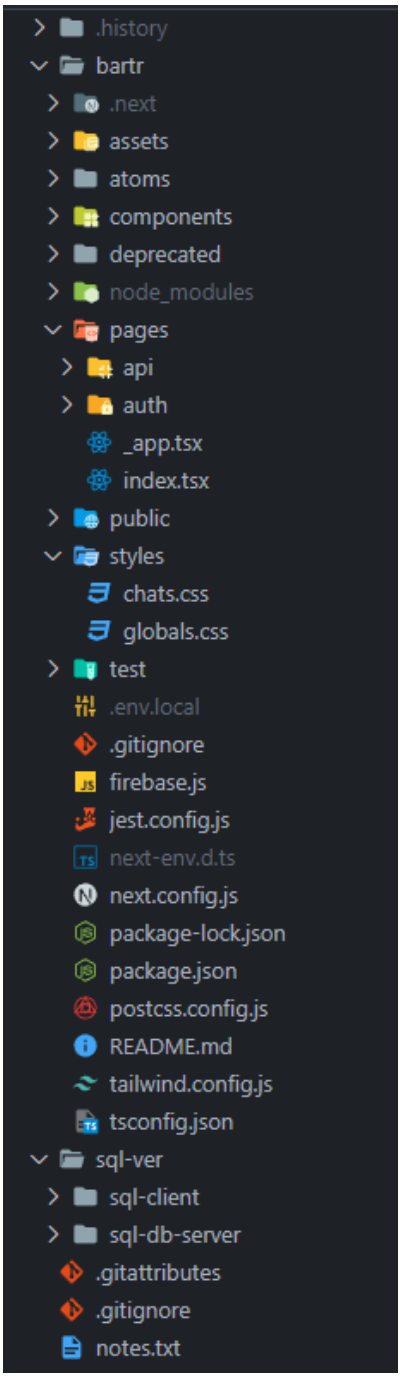
16. ábra – PC/Teljes nézet

Alapértelmezett, PC-nézetben már alapértelmezettként megjelenik a kategóriaválasztó is, ennek használatával szűkíthetők a megjelenített termékek. A képernyő jobb oldalán láthatóvá válik a profilkártya, később ez lesz az a felület, ahol szükség szerint megjelenítődnek a felhasználónak címzett rendszerüzenetek és hirdetésekkel kapcsolatos értesítések. Ezalatt található a hirdetésajánló, ahol az eddigi mentett hirdetések alapján (érték, kategória) személyre szabott hirdetések között lehet válogatni. Ezek kártyapakliként helyezkednek el, balra vagy jobbra kell húzni őket aszerint, hogy igényt szeretne-e formálni rájuk a látogató.

7.5 AZ ALKALMAZÁS FELÉPÍTÉSE, VÉGPONTOK

Ebben a fejezetben a programom azon részeit szeretném bemutatni, ami a kód megtekintése nélkül nem látható.

Maga az alkalmazás komponensekből áll, ezeket úgy hoztam létre, hogy – mivel napjainkban a technológia rohamos szinten fejlődik és bármikor megjelenhet egy olyan újítás, amit érdemes lenne integrálni a programba – köztük oda-vissza függés sose legyen. Ez később jól is jött, mert amikor local SQL adatbázisról Cloud-NoSQL-re váltottam, alig kellett néhány sort átírni. Talán érdemes még néhány szót ejteni magáról a mappaszerkezetről, amiben a fejlesztést végeztem: Itt tartottam egy rövid kis kutatást, igyekeztem a jelenleg legnagyobb releváns nyílt forráskódú Github projektek felépítéseit lemásolni. A következő oldalon látható egy ábra, ami segít vizionálni az applikáció mappastruktúráját.



17. ábra – A projekt felépítése

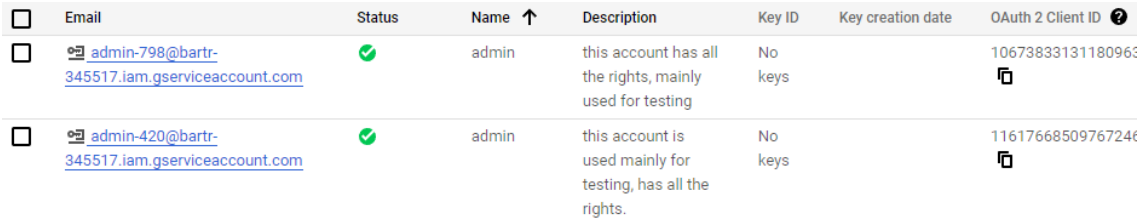
Táblázatos formában szemléltetném a kliens által használt végpontokat, jelezve azokat, amelyek csak a bejelentkezés után érhetők el. Ilyen például a hirdetésfeltöltés, vagy a saját hirdetés törlése. Nem minden végpont érhető el (Felhő-adatbázis esetén végképp) direkt módon HTTPS címről, a Firebase esetén például egy virtuális tárhely-entitás jön létre, ennek a függvényeit hívjuk és konfiguráljuk, amik magukban intézik a végpont-interakciót.

Eljárás	Elérési út	Név	Leírás	Autentikációt igényel?
POST	/auth/signin	Bejelentkezés	Egyszerű bejelentkezés (Google Oauth2)	NEM
PUT	/auth/signup	Regisztráció	Regisztráció (Googe Oauth2)	NEM
GET	/users/get/{userId}	Felhasználói adatok lekérése	Visszaadja a felhasználó olyan adatait is, amik a sütikben/sessionben nincsenek benne	IGEN
PUT	/users/modify/{userId}, {userObj}	Felhasználó adatainak módosítása	Az ID-vel megjelölt felhasználó adatait frissítjük az új objektum adataival	IGEN
GET	/items/getall	Hirdetések lekérése	Lekérjük az összes hirdetést	NEM
GET	/saveditems/getall/{userId}	Mentett hirdetések lekérése	Lekérjük a felhasználó által elmentett hirdetéseket	IGEN
POST	/items/upload	Hirdetés feltöltése	Feltöltünk egy új hirdetést	IGEN
DELETE	/items/delete/{itemId}	Hirdetés törlése	A kiválasztott hirdetést töröljük a nyilvántartásból	IGEN
GET	/offers/getallincoming/{userId}	Bejövő ajánlatok lekérése	Visszaadja a felhasználóhoz beérkező (csere)ajánlatokat	IGEN
GET	/offers/getalloutgoing/{userId}	Kimenő ajánlatok lekérése	Visszaadja a felhasználó által elküldött (csere)csereajánlatokat	IGEN
GET	/offers/getforitem/{itemId}	Adott tárgy csereajánlatainak lekérése	Visszaadja a kiválasztott tárgyra beérkezett (csere)ajánlatokat. Csak akkor használható, ha az adott felhasználó a tárgy tulajdonosa	IGEN
POST	/offers/create/{itemId}	Ajánlat küldése	Ajánlatküldés a kiválasztott hirdetésre	IGEN
DELETE	/offers/delete/{offerId}	Ajánlat törlése	Kimenő ajánlat visszavonása	IGEN
PUT	/offers/refuse/{offerId}	Ajánlat	Bejövő ajánlat	IGEN

		elutasítása	elutasítása	
PUT	/offers/accept/{offerId}	Ajánlat elfogadása	Bejövő ajánlat elfogadása	IGEN

2. táblázat – Az alkalmazás főbb végpontjai

Más hirdetés törlésére, módosítására, idegen felhasználók adatainak módosítására csak az admin szintű felhasználóknak van joga, ebből jelenleg kettő van:



18. ábra – Admin-szintű felhasználók és azok adatai

Egy kis keresgélés után felfedezhető néhány fejlesztést szintén meggyorsító, Google által kínált beépített funkció, legyen az szerveroldali titkosítás vagy akár fizetéskezelő rendszer Stripe néven. Ezek első látásra nagyon kecsegtetőnek tűnnek, viszont könnyen vendorlockot[16] idézhetnek elő. Eme funkciókból csak azokat használtam, amiknek nélkülözése még nem nehezítené meg egy esetleges más cloud-platformra (például AWS vagy Azure) való átállást.

[16] Olyan helyzet / szituáció, ahol a szolgáltatást igénybe vevő nagyon nehéz helyzetbe kerülne, ha szolgáltatót váltana

8 TESZTELÉS

Ebben a fejezetben azt fogom részletezni, hogy miként bizonyosodtam meg a webalkalmazás megfelelő és megbízható működéséről. Mivel a fejlesztés során az egyik legfőbb célom a lehető legteljesebb platformfüggetlenség és modularitás volt, a tesztelési folyamathoz is másképp kellett hozzáálljak, mint általában. Többet teszteltem a webapp elkészítésének korai fázisaiban, mert akkor történtek az igazán nagy változások, azokban a részekben szembesültem azzal, hogy ami 10 perce még működött, az már nem.

Különös figyelmet fordítottam a tartalom számomra megfelelő megjelenítésére, és arra, hogy – természetesen bizonyos keretek között – a használhatóság ne sérüljön attól, hogy néhány felhasználó csak alacsonyabb felbontású eszközökön éri el az alkalmazásomat. A túlságosan magas felbontás is lehet problémás, mert ha nem készítek minimális és maximális korlátokat a megjelenített elemek, gombok méretéhez kapcsolódóan, akkor olyan aprók lesznek, hogy a használhatóság rovására megy. Ez egyébként a Facebooknál és a Facebook Marketplace-nél jelenleg is egy valós probléma.

19. ábra – Megoldásom arra, hogy a méretezés mindig megfelelő legyen

Ahogy a mellékelt ábrán is látszik, ezt nem mindenhol volt egyszerű elintézni úgy, hogy a kód bonyolultsága ne ugorjon meg.

A saját gépemen futtatott Db-Back-end esetén a következő jelenség kisebb mértékben volt gond, de azzal kapcsolatban is próbáltam megoldást nyújtani, hogy – mivel mobileszközönél ez gyakorta megtörténhet – elkerüljem azokat a helyzeteket, amikor valami abból kifolyólag romlik el, hogy egy-egy folyamat közben elmegy az internet-kapcsolat.

A webapp korai fázisaiban még volt olyan, hogy ha a egy termék feltöltése közben megszakítom az internetkapcsolatot, az feltöltésre kerül, viszont hiányosan, feltöltött hirdetés - átlag felhasználó szemszögéből – módosítására viszont még nem volt lehetőség, kellemetlen szituációt okozott. Ezt úgy írtam át, hogy csak akkor számítson egy feltöltés sikeresnek, ha a belső ellenőrzés visszajelezte, hogy valóban megérkezett és helyére került az összes adat.

```
const uploadItem = async () => {
  if (loading) return;

  setLoading(true);

  const itemRef = await addDoc(collection(db, "items"), {
    email: session.user.email,
    owner: session.user.name,
    title: titleRef.current.value,
    value: valueRef.current.value,
    category: valueRef.current.value,
    description: descriptionRef.current.value,
    profileImg: session.user.image,
    timestamp: serverTimestamp(),
  });

  console.log("New item uploaded with the id of: ", itemRef.id);

  const imageRef = ref(storage, `items/${itemRef.id}/image`);

  await uploadString(imageRef, selectedFile, "data_url").then(
    async (snapshot) => {
      const downloadURL = await getDownloadURL(imageRef);

      await updateDoc(doc(db, "items", itemRef.id), {
        image: downloadURL,
      });
    }
  );

  setOpen(false);
  setLoading(false);
  setSelectedFile(null);
};
```

20. ábra – A hirdetés-feltöltés funkció sorrendisége

Felhőben futó Back-end esetén kevesebbszer adódott belőle probléma, végül az aszinkron függvények belső folyamatainak megfelelő átrendezése után úgy hiszem, ettől már nem kell tartanom.

Úgy gondolom, hogy az alkalmazás tesztelése során kellően sok szituációt és tényezőt figyelembe vettem, a főbb funkciók működésére külön figyelmet fordítva. Az alábbiakban megemlítenék néhány érdekesebb tesztesetet.

#	Követelmény	Tesztleírás	Mérés és tapasztalás	Eredmény
1	A felhasználó Google fiókkal be tud jelentkezni a webalkalmazásba.	A felhasználó rákattint a bejelentkezés gombra, ekkor megjelenik a szükséges felület.	A gomb megjelenik, az API működik.	✅
2	A sessionbe bekerülnek a felhasználó Google fiókjából lekért adatok	A profilkártyán megjelennek a felhasználó, sessionből kiolvasott adatai.	A kártyán az adatok mindig megjelennek, kijelentkezés után pedig eltűnnek.	✅
3	A kétlépcsős azonosítás után a látogató a főoldalra kerül.	A felhasználó belép a bejelentkezési felületen keresztül.	Új felhasználó esetén SMS vagy E-mailes hitelesítés után megtörténik a beléptetés.	✅
4	Legyen lehetőség új felhasználói fiók létrehozására.	Az Oauth API meghívódik és elindítható a folyamat.	Megnyílik a Google regisztrációs felülete és elkészíthető az új felhasználói fiók.	✅
5	Új felhasználói fiók létrehozása után a felhasználó át legyen irányítva a főoldalra.	A bejelentkezés után (maximum 5 másodperc elteltével) megtörténik a főoldalra való átirányítás.	A bejelentkezés után azonnal a főoldalra kerülünk.	✅
6	Elfelejtett jelszó esetén legyen lehetőség újat készíteni.	Az Oauth API e-mailt küld a megadott e-mail címre, ahonnan elindítható a folyamat.	Az új jelszó generáló linkre megérkezik a megadott e-mail címre.	✅
7	Kijelentkezés után ne legyenek megjelenítve az akkor már nem elérhető funkciók.	A megfelelő gombok eltűnnek, a dev tools-ból sem hívhatók elő újra.	Kijelentkezés után a dev tools-ból is eltűnnek az elrejtendő elemek.	✅

8	Az alkalmazásnak mobil-nézetbe kell lépnie, amennyiben a felbontás indokolja azt.	768px-es képernyőszélesség alá méretezzük az alkalmazás ablakát.	Az ablak átméretezésével a megfelelő nézetbe kerülünk.	✅
9	Az alkalmazásnak táblagép-nézetbe kell lépnie, amennyiben a felbontás indokolja azt.	1024px-es képernyőszélesség alá, de 768px-es szélesség fölé méretezzük a böngészőablakot.	Az ablak átméretezésével a megfelelő nézetbe kerülünk.	✅
10	Az alkalmazásnak teljes (PC) nézetbe kell lépnie, amennyiben a felbontás indokolja azt.	1024px-es képernyőszélesség fölé méretezzük a böngészőablakot.	Az ablak átméretezésével a megfelelő nézetbe kerülünk.	✅
11	Egyetlen nézetben sem takarhatják egymást a vizuális elemek	Különböző megjelenítőkön, különböző böngészőkkel ellenőrizzük a böngésző dev tools-ból.	A táblagép és PC-nézet közti váltáskor néha beleér a keresőcsík az „hirdetés létrehozása” gombba.	🆗
12	Az „új hirdetés feladása” funkció minden megadott adatot rögzít és csak a megfelelő formátumú adatot fogadja el.	Az összes adatot betáplálják és elindítják a hirdetés feltöltését.	Minden adat megfelelően rögzítésre kerül, viszont a termék értékének megadásánál nem csak számokat fogad el.	🆗
13	Új hirdetést csak akkor lehet feladni, ha minden kötelező adat ki van töltve.	A felhasználó úgy kattint a „feltöltés” gombra, hogy nem adja meg a termék nevét .	A feltöltés-gomb csak akkor válik elérhetővé, ha az összes adat meg lett adva.	✅
14	Az új hirdetés bekerül az adatbázisba.	A feltöltés után keresést indítunk a termék nevére az adatbázisban.	A feltöltött termék megjelenik.	✅
15	Az adatbázisban való	Törlünk egy	A termék a Front-	✅

	változás azonnal megjelenik a webalkalmazásban, a Virtuális DOM megfelelő része automatikusan frissül.	hirdetést az adatbázisból, Back-end oldalon.	enden is azonnal eltűnik.	
16	Ajánlatküldés esetén a tárgy tulajdonosa azonnal értesítést kap.	A bejelentkezett látogató megnyomja az „ajánlatküldés” gombot.	Az értesítés megjelenik.	✅
17	A hirdetés tulajdonosa nem tud ajánlatot küldeni a saját termékére	Az ajánlatküldés gombja meg sem jelenik.	Az ajánlatküldés gomb nem tűnik el saját termékeknél.	❌
18	A személyre szabott ajánlatok paklijai megfelelően jelenítik meg a különböző méretarányú képeket.	A személyre szabott ajánlatok maguktól legenerálódnak és hiba nélkül megjelennek.	Kizárólag egészen extrém méretarányú képeknél problémás a megjelenítés, az viszont elfogadható.	✅
19	A tárgyak hirdetései alatt lévő üzenetküldés funkció működik.	A felhasználó begépel egy üzenetet és entert nyom, vagy a „küldés” gombra kattint.	Az üzenet elküldése megtörténik.	✅
20	Ha a felhasználó üzenetet kap, arról azonnal értesítést kap.	A bejelentkezett látogató navigációs sávján piros háttérrel megjelenik az olvasatlan üzenetek száma.	Az értesítés megjelenik.	✅
21	Az új, még meg nem tekintett üzenetek és csereajánlatok megjelennek a navigációs sávon.	A bejelentkezett látogató navigációs sávján ahogy az olvasatlan üzenetek, az új ajánlatok is megjelennek.	Az értesítések frissítés nélkül is láthatóak.	✅

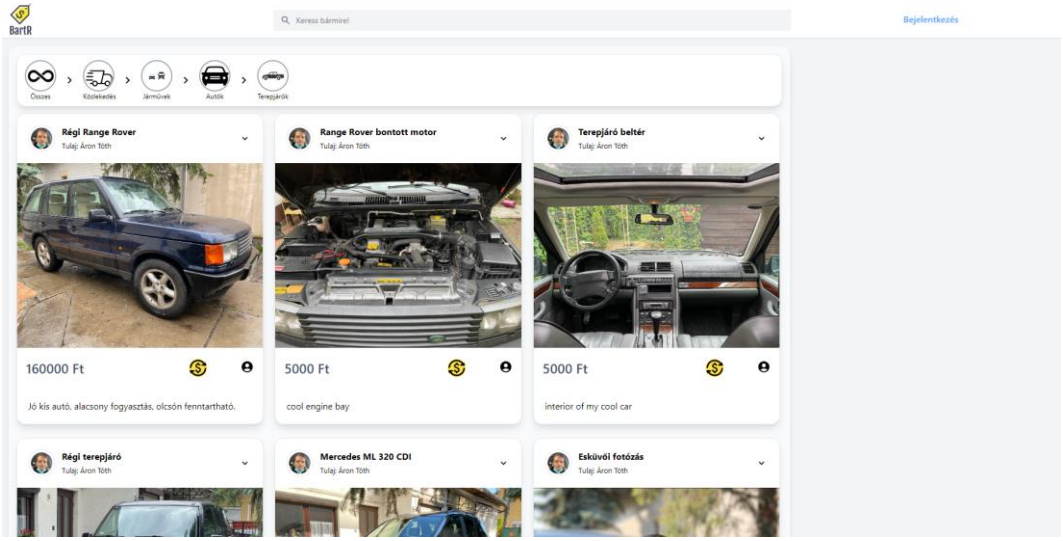
22	A hirdetések kategorizálása megfelelően működik.	A kategóriákat böngészve csak az abban szereplő termékek kerülnek megtekintésre.	A kategorizálás Back-end oldalon megfelelő, Front-enden még nincs hozzáigazítva a megjelenítés.	✗

3. táblázat – Tesztelések

Amint látszik, nem minden teszt lett 100%-ban sikeres, de talán ez is bizonyítja, milyen fontos dolog a tesztelés. A talált hibák a programom jövőbeli verzióiban természetesen javításra fognak kerülni.

9 FELHASZNÁLÓI DOKUMENTÁCIÓ

Az elkészült alkalmazásomat (Egy esetleges deployment[17] után) bárki elérheti, az oldal látogatásához felhasználói autentikáció nem szükséges. Ennek okát a piackutatás fejezében már taglaltam.

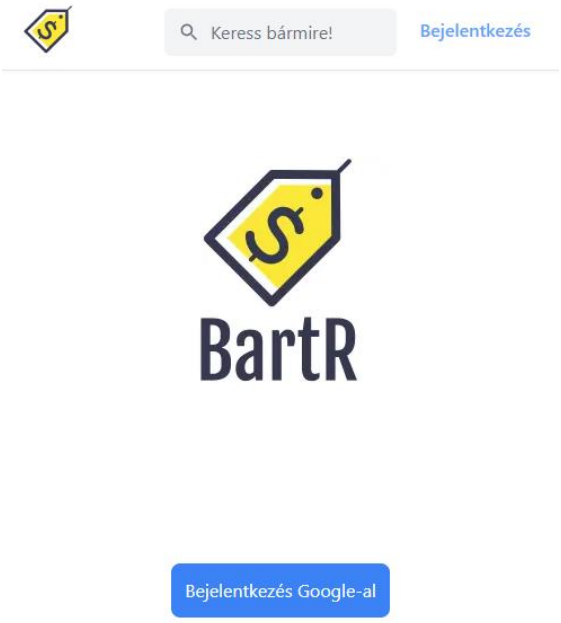


21. ábra – Bejelentkezés előtti állapot

Ahogy a képernyőképen is látszik, a hirdetések böngészésén kívül bejelentkezés-nélküli állapotban a hirdetések böngészésén kívül nem sok mindenre van lehetőség.

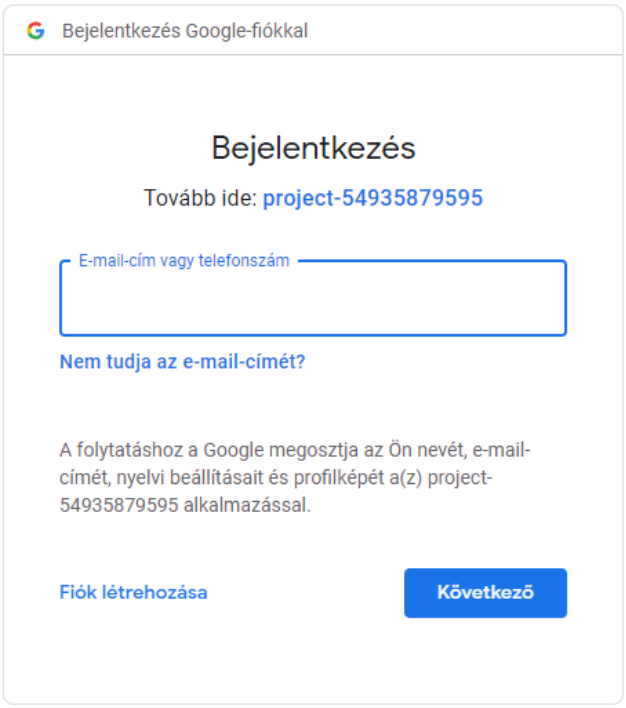
A főbb funkciók nagyrészéhez már bejelentkezés szükséges, ez a Google Oauth többfaktoros autentikációjával lehetésges, amit a navigációs sáv (itt jelennek meg az olyan értesítések, mint a beérkező üzenetek vagy ajánlatok, innen kereshetünk és itt tudunk visszalépni a főoldalra is, a BartR ikonra kattintva) jobb szélén lévő „Bejelentkezés” gombra kattintva érhetünk el.

[17] Az alkalmazást egy felhőszolgáltatásba installáljuk, így akkor is elérhető lesz, ha kikapcsoljuk a saját számítógépünket



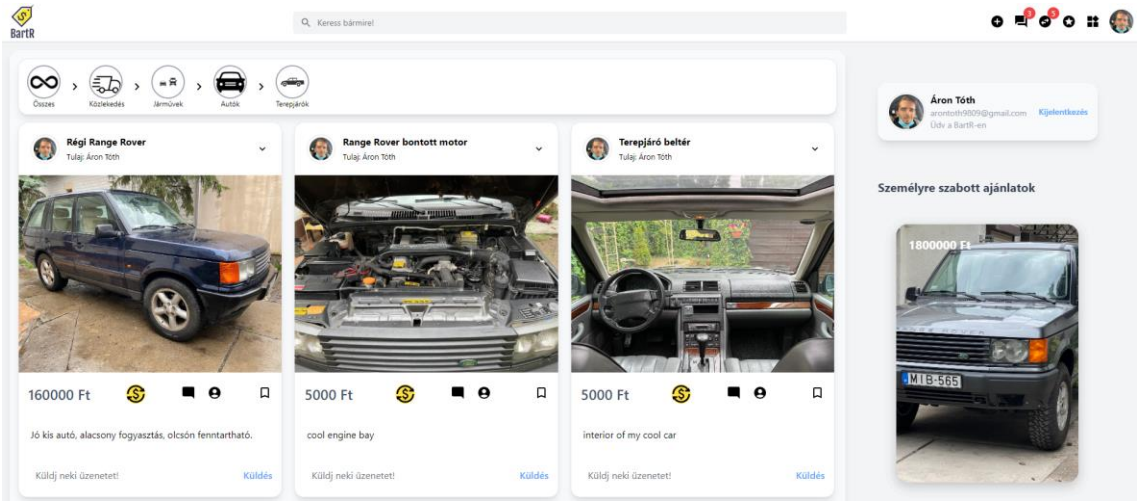
22. ábra – A bejelentkezési képernyő

Ha a webalkalmazást később további bejelentkezési lehetőségekkel szeretném majd bővíteni, a „Bejelentkezés Google-al” gomb alatt fognak megjelenni a további gombok. A Facebookos hitelesítés lehetőségét egy másik verzióban már előkészítettem. Az Oauth általi bejelentkezéshez felhasználónév-jelszó párost csak akkor kell megadni, ha az adott böngészőben egyáltalán nem vagyunk belépve egy Google fiókba sem. Ellenkező esetben csak engedélyt kell adni az alkalmazásnak, hogy hozzáférjen a működéshez szükséges alapadatokhoz. Ezeket igyekeztem úgy megválogatni, hogy a potenciális látogatók ne lássanak informatikai biztonsági kockázatokat az adataik kiadásában.



23. ábra – Bejelentkezés, ha a böngészőben még nincs aktív Google felhasználó

Autentikáció után a következő látvány fogad minket: Elérhetővé válnak az eddig rejtett funkciók, mint a személyre szabott hirdetések, üzenetek vagy ajánlatok fogadása és küldése.

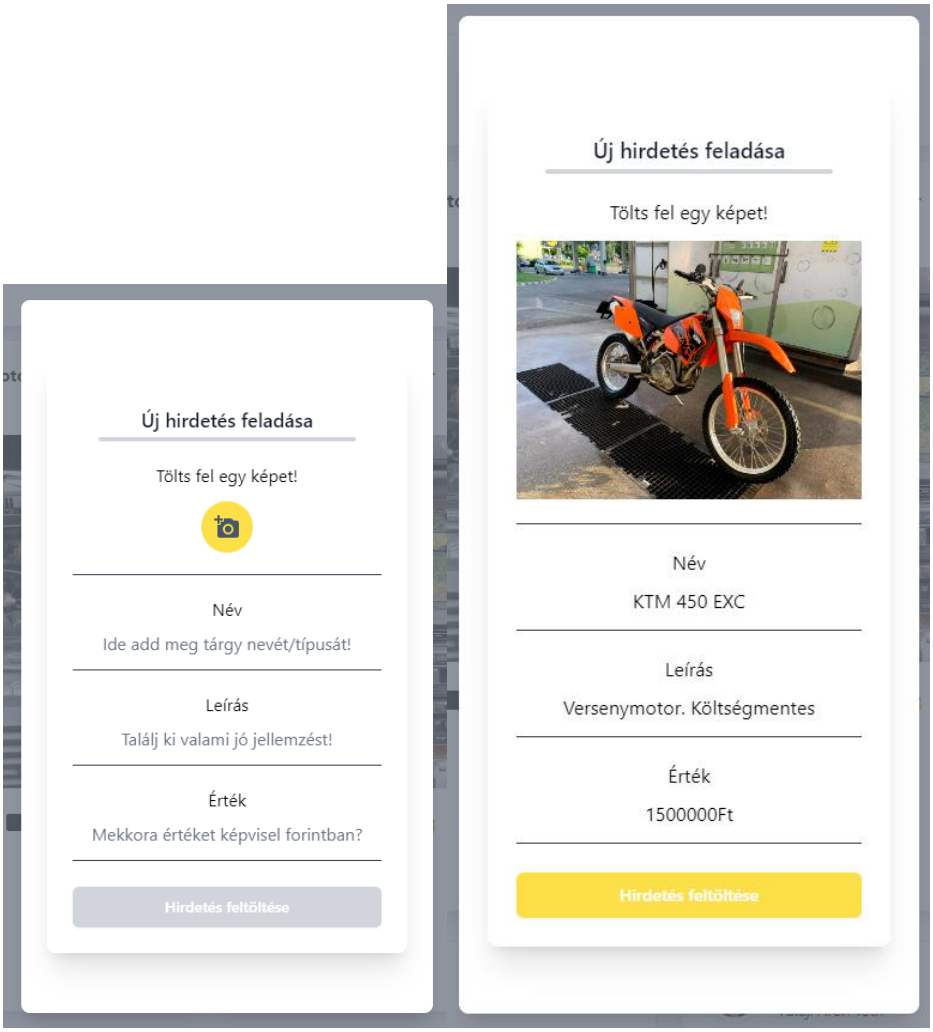


24. ábra – Bejelentkezés után az összes funkció elérhető

A személyre szabott hirdetések között úgy tudunk navigálni, hogy Tinder-szerűen balra-jobbra húzással „ledobjuk” őket a képernyőről. Ha bármelyik termékre visszük az egeret

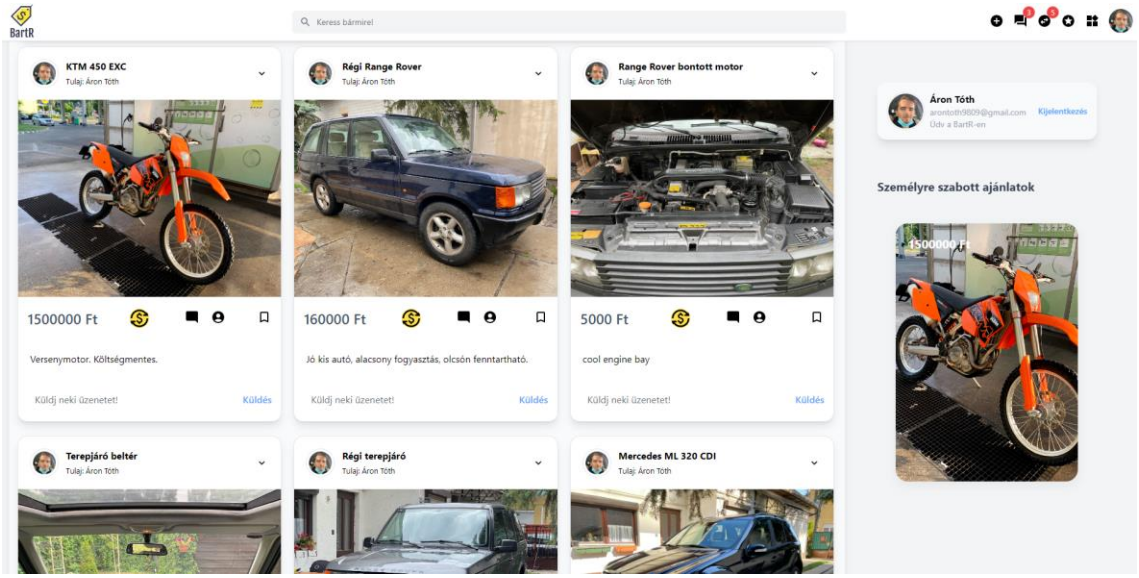
(vagy mobilon és tableten az ujjunkat), az 105%-os méretűre nő, ezzel is könnyítve a részletek megtekintését.

A navigációs sáv jobb szélén jobbról balra haladva a következő funkciókat érhetjük el: Saját profil megnyitása, felhasználó összes hirdetésének megtekintése, mentett hirdetések megtekintése, beérkező ajánlatok megtekintése, beérkezett üzenetek megtekintése, új hirdetés feltöltése. Utóbbira kattintva egy ablak ugrik fel, a képernyő többi részét elsötétítve.



25. ábra – Új hirdetés feladása - adatok kitöltése előtt és után

Ebben az ablakban kötelezően ki kell választanunk legalább egy feltöltendő képet a cserére/eladásra bocsátandó tárgyról, majd megadni a nevét, egy rövid (de akár hosszú is lehet, maximális hossza csak a Back-enden van limitálva) leírást és egy becsült értéket, ami a többi látogató számára is segít beazonosítani, mivel számoljanak, ha hozzá szeretnének jutni. Mindezek után a feltöltés-gomb elérhetővé válik és egy kattintás hatására néhány pillanaton belül meg is jelenik az apróhirdetésünk a többi között.



26. ábra – Az új hirdetés frissítés nélkül megjelenik

Ha valami megtetszik, az árától jobbra lévő csere ikonra kattintva jelezhetjük az eladónak szándékunkat, vagy üzenetet küldhetünk neki az alatta található sávba való gépeléssel és az enter lenyomásával, vagy a „küldés”-re való kattintással.

10 KONKLÚZIÓ

A fejlesztési folyamat során jónéhány olyan problémát találtam és küzdöttem le, melyről azt gondolom, hogy érdemes lenne említést tennem róla. Ezekről szeretnék beszélni az elkövetkező fejezetben.

Az egyik a technológiaválasztás. Mivel ebből a programból nem csak sikeres szakdolgozatot, hanem üzletet is szeretnék csinálni, olyan technológiákat szerettem volna használni, amik jövőállóak és kompromisszummentesek. Ez azzal járt, hogy több helyen el kellett engednem a könnyebb megoldást és helyette olyat választani, amivel ahhoz, hogy dolgozhassak, lényegesen több időt kell belefektessek. A React + Next.JS végül úgy látszik, hogy jobb választásnak bizonyult az Angularral szemben, mert rengetegszer tudtam hasznát venni a Virtual DOM-nak, ami ugye kizárólag React-tal elérhető.

Sokat gondolkodtam a mobilos alkalmazás elkészítésének módján és folyamatán is. Itt kellő piackutatás után arra jutottam, hogy érdemesebb lenne az alap webalkalmazást úgy elkészíteni, hogy platformfüggetlen legyen, sőt lehetőleg kihasználja az épp aktuális megjelenítők tulajdonságait (méretezés, felbontás, egérrel vagy érintőképernyővel történő vezérlés) is. Ebben a Tailwind nagy segítség volt, mert lehetőséget adott arra, hogy pixelekre bontva konfigurálhassam a webapp kinézetét. Egy másik, nem elhanyagolható jelenség az, hogy (Nem kell sokáig keresni, vegyük akár a Jófogás vagy a Craigslist, esetleg Vatera példáját) ha különválasztjuk a web és -mobilalkalmazást, azokat külön, párhuzamosan kell fejleszteni. Ha csak a három fő platformot céloznám meg (Windows, Android, IOS/MAC), már akkor is három teljesen különböző alkalmazást kellene párhuzamosan fejleszteni, ez még a legnagyobb cégeknek sem megy gördülékenyen. Így viszont, hogy egy fő alkalmazást ruházok fel platform-függően különböző tulajdonságokkal, lehetőségem van mindhárom (Sőt, ugyanúgy elfut Linuxon, vagy szinte bármilyen rendszeren, ahol elérhetőek a modernebb böngészők) rendszert egyformán „támadni”.

Próbáltam minél több rendszert és lehetőséget kipróbálni és ezeket beépíteni a szakdolgozatomba is, mert úgy hiszem, hogy nem csak a végeredmény számít, hanem maga a folyamat is, amin keresztül eljutottam ide. Törekedtem arra, hogy ne csak papíron legyen moduláris az alkalmazás és ha a záróvizsga után fél évvel úgy döntök, hogy kipróbálnám az Azure szolgáltatásait és azokat használnám a Back-end alapjául, meg tudjam tenni anélkül, hogy szignifikáns időt kelljen eltöltenem a program erre való felkészítésével.

11 TOVÁBBFEJLESZTÉSI LEHETŐSÉGEK

Ebben a fejezetben azokról a dolgokról és további funkciókról írok, amik a szakdolgozatom részét már nem képezik, de úgy gondolom, hogy ha igazán komoly és piacképes alkalmazást szeretnék létrehozni, a jövőben érdemes lenne időt tölteni az implementálásukkal.

Ilyen például a chat-funkció továbbfejlesztése. Jelenlegi formájában működőképesnek mondható, de használata körülményes és véleményem szerint még nem eléggé intuitív. Mivel egy harmadik osztálytól származó könyvtárat használok hozzá (ChatEngine), a felhasználói fiókok Google Oauth-tal való szinkronizálása nem zökkenőmentes. Jelen pillanatban úgy látom, hogy a legbölcsebb megoldás egy saját felület létrehozása lenne, az viszont – ha azt akarom, hogy valóban úgy működjön, ahogy megálmodtam - olyan komplexitású munka lenne, ami talán egy külön szakdolgozat alapjául is szolgálhatna.

Akad egy másik, valós probléma is. Mivel az alkalmazásom böngészőből fut, egy esetleges üzenet vagy csereajánlat beérkezéséről a látogató csak akkor értesül, ha meg van nyitva az oldal és rápillant a navigációs sávra. Nagyon hasznos funkció lenne az is, ha ezeket az értesítéseket valamilyen módon úgy is el tudnánk juttatni a felhasználónak, hogy közben az alkalmazás meg sincs nyitva. Erre megoldás lehet a böngészők API-jainak mélyebb megismerése utáni azonnali értesítések kiépítése, vagy az e-mailes értesítés. Ezutóbbit járható útnak tartom, mert korábban már kísérleteztem ezzel és értem el sikereket. Itt viszont kihívás tud lenni az e-mail szolgáltató kiválasztása is, mert nagyon nagy szórása van az általuk kínált funkcióknak.

Éles környezetben elkerülhetetlen téma lenne a monetizáció is. Ez a kérdéskör külön fejezeteket tölthetne be, mert nagyon sok – egyébként kiváló alkalmazás – itt bukott el. Úgy hiszem, járható út lenne a reklámok bevezetése lenne oly módon, hogy a tartalom megjelenítését és maga a funkcionalitást ne rontsák túlságosan. A Google erre is kínál házon belüli megoldást; ott van a Google Ads, az AdSense, az Analytics, viszont nem tanácsos ennyire elköteleződni egy-egy szolgáltató mellett, ugyanis ez szintén vendorlockhoz vezet. Az Amazon szintén kiváló eszközöket tud biztosítani, habár az Ő árazásukkal nem vagyok teljesen kibékülve.

12 ÖSSZEGZÉS

A dolgozatom célja az volt, hogy egy saját projekt keretein belül egyesítsem a jelenleg fellelhető legsikeresebb és legérdekesebb online kereskedelmi platformok előnyeit, ezzel együtt pedig kiszűrjem a bennük felfedezett hiányosságokat, hibákat. Úgy hiszem, hogy ez sikerült. Emellett még tervezési folyamat elején elhatároztam azt is, hogy olyan szoftvert szeretnék összerakni, ami nem csak a szakdolgozatom megírásához járul hozzá, hanem megfelelő alapja lehet annak a terméknek, amiből később (remélhetőleg) jövedelmező üzletet fogok csinálni.

Az irodalomkutatásom során igyekeztem minél több olyan apróságot megtalálni, ami miatt nem csak papíron, de a valóságban is vonzó lehet a koncepcióm a célközönség számára, emellett elkerülni azokat a hibákat, melyek elkövetése azt eredményezné, hogy a projektem csak egy újabb bukott próbálkozás legyen a többi platform között.

Amikor belekezdtem a fejlesztésbe, az egyik legnagyobb félelmem az volt, hogy nincs elég tudásom az általam megálmodott állapot alkalmazás összerakásához. Ez így is volt, viszont az elém gördülő akadályokat le tudtam küzdeni, amire büszke vagyok. Bár az eredmény messze nem tökéletes, már látom az irányt, amerre tovább fogok haladni és olyan tapasztalatokat szereztem, amiket a szakmai karrierem során is hasznosíthatok majd.

A feladatleírás minél teljesebb megvalósítása mellett próbáltam hasznos tudásra szert tenni az új technológiák megismerése által. Aminek külön örülök, hogy megláttam a szépséget a Javascriptben és a Reactban, de rájöttem arra is, hogy mekkora potenciál rejlik a felhőszolgáltatásokban. Véleményem szerint hasznos megközelítés volt a többféle operációs rendszeren és platformon való fejlesztés, ebből kifolyólag a tesztelésnél lehetőségem adódott teljesen különböző környezetekben vizsgálni az alkalmazás működését.

13 SUMMARY

The goal of my thesis was to combine the advantages of the most successful and interesting online trading platforms currently available within the framework of my own project, and at the same time to filter out the shortcomings and errors discovered in them.

I think it worked. In addition, even at the beginning of the planning process, I decided that I would like to put together a software that would not only contribute to the writing of my thesis, but could also be a suitable basis for the product that I will later (hopefully) turn into a profitable business. During my research, I tried to find as many things as possible, which would make my concept attractive to the target audience not only on paper, but also in reality. I did try everything to avoid those mistakes, the making of which would result in my project being just another failed attempt like the ones I wrote about.

When I started development, one of my biggest fears was that I didn't have enough knowledge to put together the kind of application I desired. That was as it was, but I was able to overcome the obstacles that came my way, which I am proud of. Although the result is far from perfect, I can already see the direction in which I will continue and I have gained experiences that I will be able to use in my professional career.

In addition to implementing the task description as completely as possible, I tried to gain useful knowledge by getting to know new technologies. What I'm especially happy about is that I saw the beauty in Javascript and React, but I also realized how much potential there is in cloud services. In my opinion, developing on multiple operating systems and platforms was a useful approach, as a result of which I had the opportunity to test the operation of the application in completely different environments.

12 IRODALOMJEGYZÉK

[1] Historyworld.net [Hivatkozva: 2021.10.02] http://www.historyworld.net/wrldhis/PlainTextHistories.asp?ParagraphID=bfy

[2] Relational vs NoSQL data [Hivatkozva: 2021.12.01] https://docs.microsoft.com/en-us/dotnet/architecture/cloud-native/relational-vs-nosql-data

[3] React Native vs Flutter: What is better for app development in 2021? [Hivatkozva: 2021.12.01] https://nix-united.com/blog/flutter-vs-react-native/

13 ÁBRAJEGYZÉK

14 TÁBLAJEGYZÉK